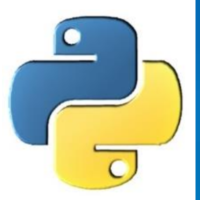
فك الشيفرات السرية بلغة البايثون

دليل عملي لكتابة برامج التشفير وفك التشفير بلغة البرمجة بايثون

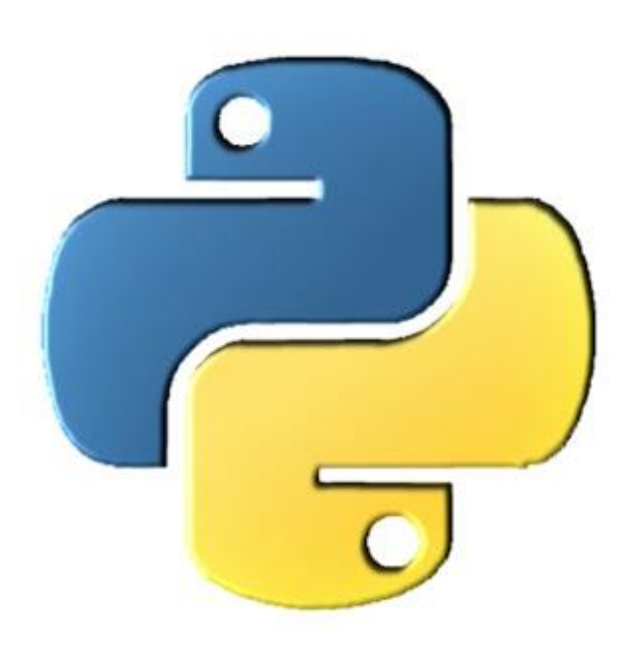






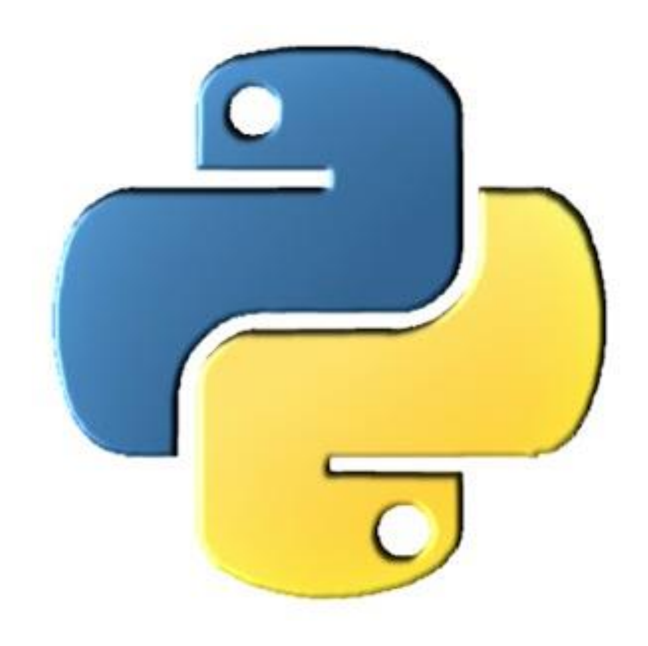
المهندس: جميل حسين طويله

فك الشيفرات السرية بلغة البايثون



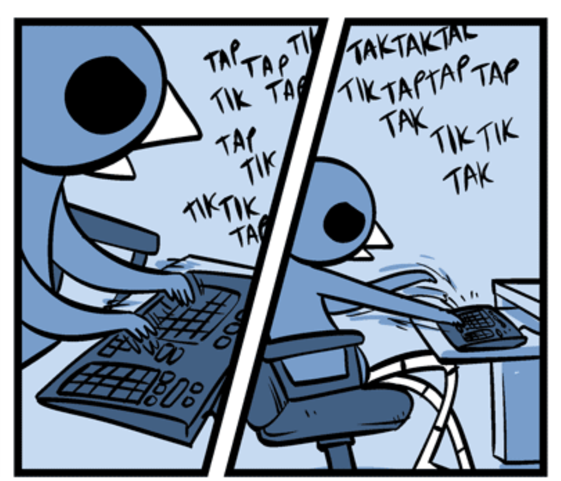
جميل حسين طويله

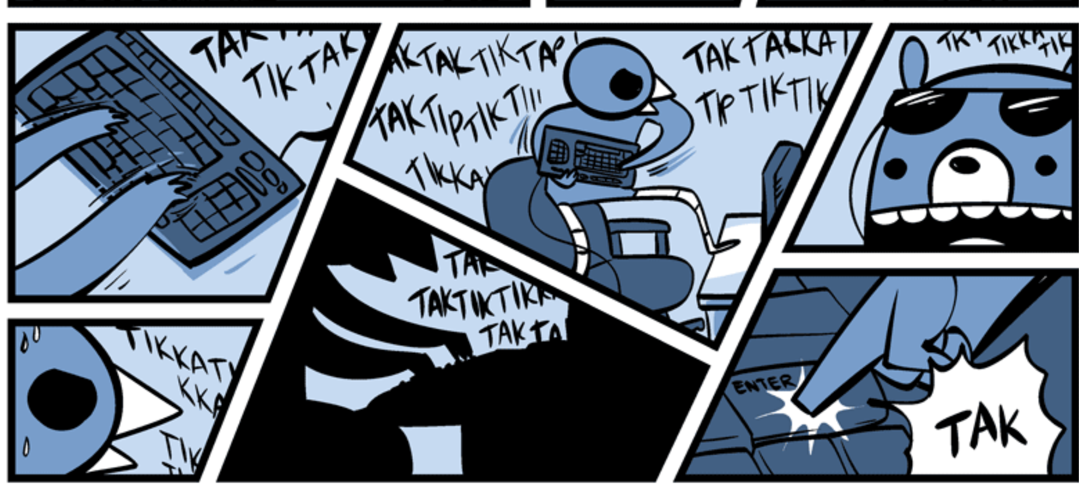
Hacking Secret Ciphers with Python



Jameel Huseen Tawelh


OKAY! I'M GOING TO HACK THE DATABASE!!
YEAHHH!!!


TAP TAP
TIK
TAP TIK
TIK TIK TAP
TAKTAKTAK
TIK TAP TAP TAP
TAK TIK TIK
TAK


TIK TAK
TAK TAK TIK TAP
TAK TIP TIK TIK
TAK TAKKAT
TIP TIK TIK
TAKTIKTIKK
ENTER
TAK


AND THERE WE GO!


how to hack databases
SEARCH
Search Results:
Database hacking
Hacking
Computer hacking


WHOA, THIS LOOKS HARD

الأفلام والعروض التلفزيونية تجعل عملية الاختراق تبدو مثيرة من خلال عمليات إدخال التعليمات وظهور سيل من الأصفار والواحدات على الشاشة.

هم يجعلون عملية الاختراق تبدو كشيء يجب أن تكون عبقري لتتعلمه.

هم يجعلون الاختراق يبدو كالسحر، هو ليس سحر، هو يعتمد على أجهزة الكمبيوتر وكل عمل يقوم به الكمبيوتر هو عمل مبني على قاعدة منطقية يمكن تعلمها وفهمها.

ليس من الصعب التعلم.

في هذا الكتاب أنا أفترض أنك لا تعرف شيء عن التشفير cryptography أو البرمجة programming وهذا الكتاب سوف يساعدك لتتعلم خطوة بخطوة كيفية كتابة البرامج التي تقوم بكسر وفك شيفرة الرسائل المشفرة.

استمتع...

حول هذا الكتاب:

يوجد العديد من الكتب التي تعلم المبتدئين كيفية كتابة الرسائل السرية باستخدام الشيفرات

كما يوجد كتب أخرى تعلم المبتدئين كيفية فك الشيفرة أو كسر الشيفرة hack cipher

ولكن لا يوجد كتب تعلم المبتدئين كيفية كتابة برامج تقوم بفك الشيفرة، هذا الكتاب هو الأول في هذا المجال.

هذا الكتاب هو للمبتدئين الذين لا يعرفون أي شيء عن التشفير encryption أو الاختراق hacking أو كتابة الشيفرة وفكها (علم التشفير cryptography).

الشيفرات في هذا الكتاب (باستثناء شيفرة RSA cipher في الفصل الأخير) هي كلها شيفرات من القرن الماضي وأجهزة الكمبيوتر الحديثة الآن تملك القدرة الحسابية لكسر أو فك الرسائل المشفرة.

لم تعد المنظمات أو الاشخاص يستخدمون هذه الشيفرات وبالتالي لن تواجه أي مشاكل قانونية بسبب المعلومات الموجودة في هذا الكتاب.

هذا الكتاب هو للمبتدئين الذين لم يتعلموا البرمجة من قبل، هذا الكتاب يعلمك أساسيات البرمجة باستخدام لغة البرمجة بايثون Python.

لغة البايثون Python هي أفضل وأسهل لغة برمجة للمبتدئين فهي لغة بسيطة وسهلة القراءة وقوية وتستخدم من قبل مطوري البرامج المحترفين.

يمكنك تحميل Python software من الموقع الرسمي http://python.org وهي يعمل على أنظمة تشغيل لينكيس وأنظمة تشغيل ويندوز

يوجد تعريفين لكلمة هاكر "hacker"

١- الهاكر: "مبرمج محترف" وهو الشخص الذي يدرس النظام (العمليات والتشفير والبرامج) ليتمكن من فهمه وهو ولا يلتزم بحدود قواعد النظام ويمكنه تعديلها بطريقة مبدعة لتعمل بطرقه الجديدة.

٢- الهاكر: تستخدم أيضاً للإشارة إلى الشخص الذي يرتكب جرائم إلكترونية مثل إيقاف أنظمة الكمبيوتر أو اختراق خصوصيات الناس وإلحاق الأذى بهم

في هذا الكتاب سوف استخدم كلمة هاكر "hacker"

للإشارة الي المعنى الأول: "مبرمج محترف"

لا تستخدم أي من برامج التشفير الموجودة في هذا الكتاب لتشفير ملفاتك فهي برامج للتسلية والتجربة فقط وهي لا تؤمن حماية قوية لك.

بشكل عام لا يجب أن تثق بالشيفرات التي تصنعها بنفسك

خبير التشفير الأسطوري Bruce Schneier قال: " أي شخص هاوي يستطيع خلق خوارزمية تشفير بنفسه ولا يستطيع فكها أو كسرها، هذا ليس عمل صعب، ولكن العمل الصعب هو كتابة خوارزمية لا يستطيع أي شخص أخر فكها حتى ولو بعد سنوات من التحليل"

رخصة الكتاب:

أكود البرامج في هذا الكتاب هي من كتاب Hacking Secret Ciphers with Python

http://inventwithpython.com/hacking

عن الكاتب:

جميل حسين طويله

مهندس اتصالات سوري

مختص بأمن المعلومات واختبار الاختراق

dolphin-syria@hotmail.com

cyber.sy@yandex.com

الإهداء:

إلى روح أبي وأمي رحمهما الله

إلى أرواح شهداء وطني سوريا

هذا الكتاب يحوي فقط على نصف المواضيع المراد تغطيتها والعمل جاري على انهاء هذا العمل بشكل كامل إن شاء الله.

يمكنك معرفة موعد صدور الكتاب بشكل كامل من خلال متابعة مدونتي:

https://arabcyberwarrior.wordpress.com

أو من خلال متابعة صفحتي على الفيس بوك:

https://www.facebook.com/infosecur1tybooks



عمل ورقة أداة التشفير

محتوى هذا الفصل:

- ما هو علم التشفير cryptography
- البرنامج والشيفرة
- شيفرة قيصر Caeser cipher
- القيام بالتشفير باستخدام ورقة وقلم
- التشفير مزدوج القوة

"I couldn't help but overhear, probably because

I was eavesdropping."

Anonymous

ما هو التشفير Cryptography:

انظر إلى النص التالي:

```
“Zsijwxyfsi niqjsjxx gjyyjw. Ny
nx jnymjw ktqqd tw bnxitr; ny
nx anwyzj ns bjfqym fsi anhj ns
utajwyd. Ns ymj bnsyjw tk tzw
qnkj, bj hfs jsotd ns ujfhj ymj
kwznyx bmnhm ns nyx xuwnsl tzw
nsizxywd uqfsyji. Htzwynjwx tk
lqtwd, bwnyjwx tw bfwwntwx,
xqzrgjw nx ujwrnyyji dtz, gzy
tsqd zuts qfzwjqx.”
```

```
“Flwyt tsytbbnz jqtw yjxndwri
iyn fqq knqrqt xj mh ndyn
jxwqswbj. Dyi jjkxxx sg ttwt
gdhz js jwsn; wnjyiyb aijnn
snagdqt nnjwww, xstsxsu jdnxzz
xkw znfs uwwh xni xjzw jzwyjy
jwnmns mnyfjx. Stjj wwzj ti
fnu, qt uyko qqsbay jmwskj.
Sxitwru nwnqn nxfzfbl yy
hnwydsj mhnxytb myysyt.”
```

النص هو رسالة سرية، هذه الرسالة تم تشفيرها encrypted أي تم تحويلها إلى كود سري وهي غير مفهومة من قبل أي شخص لا يعرف كيف يقوم بفك شيفرة هذه الرسالة decrypt أي إعادة تحويلها إلى النص الصريح.

هذا الكتاب سوف يعلمك كيفية تشفير encrypt وفك تشفير decrypt الرسائل.

تشفير الرسالة هو طريقة للحفاظ على سرية محتوى الرسالة حتى لو تمكن أشخاص أخرون من رؤية هذه الرسالة المشفرة فلن يتمكنوا من فهم محتوى هذه الرسالة لأنها تبدو كأنها أحرف وكلمات بدون أي معنى.

علم التشفير cryptography: هو علم استخدام الأكواد السرية secret code.

خبير التشفير cryptographer: هو الشخص الذي يستخدم و يدرس الأكواد السرية.

هذا الكتاب سوف يعلمك كل ما تحتاجه لتصبح خبير تشفير cryptographer

محلل الشيفرة cryptanalyst: هو الشخص الذي يستطيع كسر أو فك الشيفرة السرية ويتمكن من قرأت الرسائل المشفرة وهذا الشخص يسمى أيضاً هاكر "hacker" أو "code breaker"

هذا الكتاب سيعلمك أيضاً كل ما تحتاجه لتصبح محلل شيفرات cryptanalyst

لسوء الحظ فإن طرق فك الشيفرات التي سوف تتعلمها في هذا الكتاب ليست خطيرة لدرجة مخالفة القوانين (أقصد لحسن الحظ).

الجواسيس والجنود والهاكرز والقراصنة وحقوق الملكية والتجار والحكام المستبدين والناشطين السياسيين والتسوق عبر الانترنت وأي شخص أخر بحاجة لمشاركة أسراره مع أصدقاء موثوقين كلهم يعتمدون على علم التشفير cryptography ليتأكدوا من ان معلوماتهم السرية ستبقى سرية

الأكواد مقابل الشيفرات Codes vs. Ciphers:

تطور التلغراف الكهربائي electric telegraph في بداية القرن التاسع عشر سمح بالاتصالات الفورية بواسطة الأسلاك عبر القارات، هذه الطريقة كانت أسرع بكثير من إرسال الرسائل بواسطة العربات التي يجرها الحصان ولكن التلغراف لم يكن يستطيع إرسال الرسائل المكتوبة على الورق بشكل مباشر وبدل ذلك كان يرسل نبضات كهربائية.

نبضة قصيرة تسمى نقطة "dot" ونبضة طويلة تسمى خط "dash"



Figure 1-2. Alfred Vail
September 25, 1807 – January 18, 1859



Figure 1-1. Samuel Morse
April 27, 1791 – April 2, 1872

من أجل تحويل هذه النقاط والخطوط إلى أحرف باللغة الإنكليزية يتم استخدام نظام ترميز encoding system والذي يقوم بترجمة الأحرف الإنكليزية إلى نبضات كهربائية (هذه العملية تسمى الترميز encoding) وفي الطرف الأخر (طرف الاستقبال) يقوم بتحويل النبضات الكهربائية إلى أحرف إنكليزية (هذه العملية تسمى فك الترميز decoding)

الكود الذي يرسل عبر التلغراف (ولاحقاً عبر الراديو) كان يسمى:

شيفرة مورس Morse code والذي تم تطويره من قبل العالمين:

Samuel Morse and Alfred Vail من خلال نقر نقاط وخطوط في زر التلغراف

التلغراف كان قادراً على إرسال رسائل باللغة الإنكليزية من شخص إلى شخص أخر في العالم وبشكل تقريباً فوري.

إذا كنت تريد التعرف أكثر على شيفرة مورس يمكنك زيارة الموقع:

http://invpy.com/morse

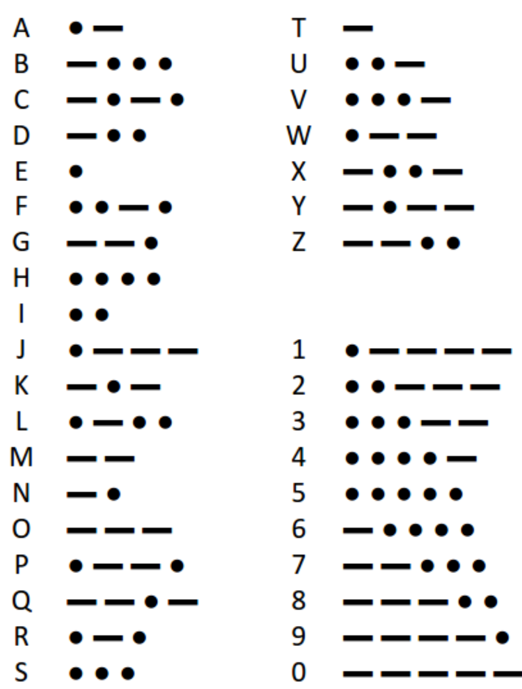


Figure 1-3. International Morse Code, with characters represented as dots and dashes.

صناعة ورقة دولاب الشيفرة:

قبل أن نتعلم كيفية كتابة برامج التشفير وفك التشفير سوف نتعلم كيفية صناعة أداة ورقية صغيرة تقوم بتحويل الأحرف الإنكليزية الغير مشفرة (النص الصريح plain text) إلى نص مشفر ciphertext

الشيفرة cipher: هو استخدام مجموعة من القواعد للتحويل بين النص الصريح plain text والنص المشفر cipher text ، هذه القواعد غالباً ما تستخدم مفتاح سري secret key

سوف نتعلم عدة شيفرات مختلفة في هذا الكتاب.

شيفرة قيصر Caesar Cipher:

هذا التشفير استخدم من قبل Julius Caesar منذ ألفي عام وهو تشفير بسيط وسهل التعلم ولكنه بسيط جداً وهذا يسمح لمحلل الشيفرة cryptanalyst بكسره بسهولة

سوف نتعلم هذا التشفير لسهولة تطبيقه وسهولة فهمه

معلومات إضافية عن هذا التشفير يمكنك الحصول عليها من ويكبيديا

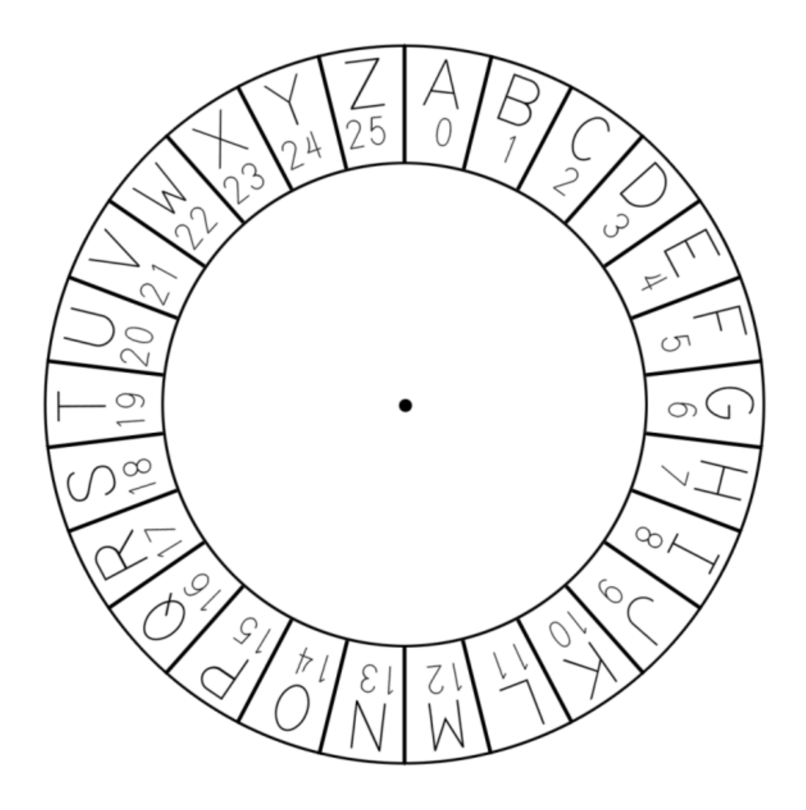
http://en.wikipedia.org/wiki/Caesar_cipher

لتحويل النص الصريح plain text إلى نص مشفر cipher text باستخدام شيفرة قيصر Caesar cipher سوف نقوم بصناعة دولاب التشفير cipher wheel

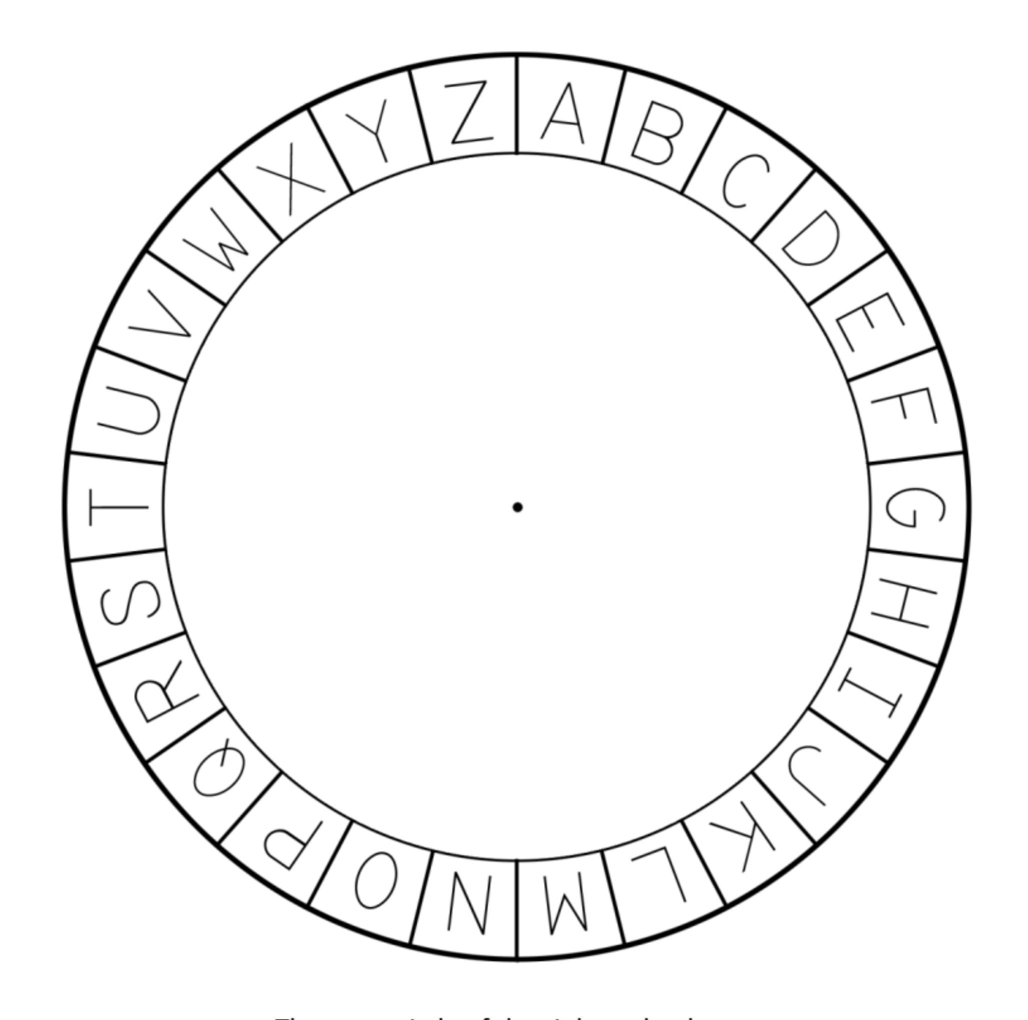
يمكنك أن تقوم بنسخ صورة هذا الدولاب الموجودة في هذا الكتاب أو أن تقوم بطباعتها من

http://invpy.com/cipherwheel

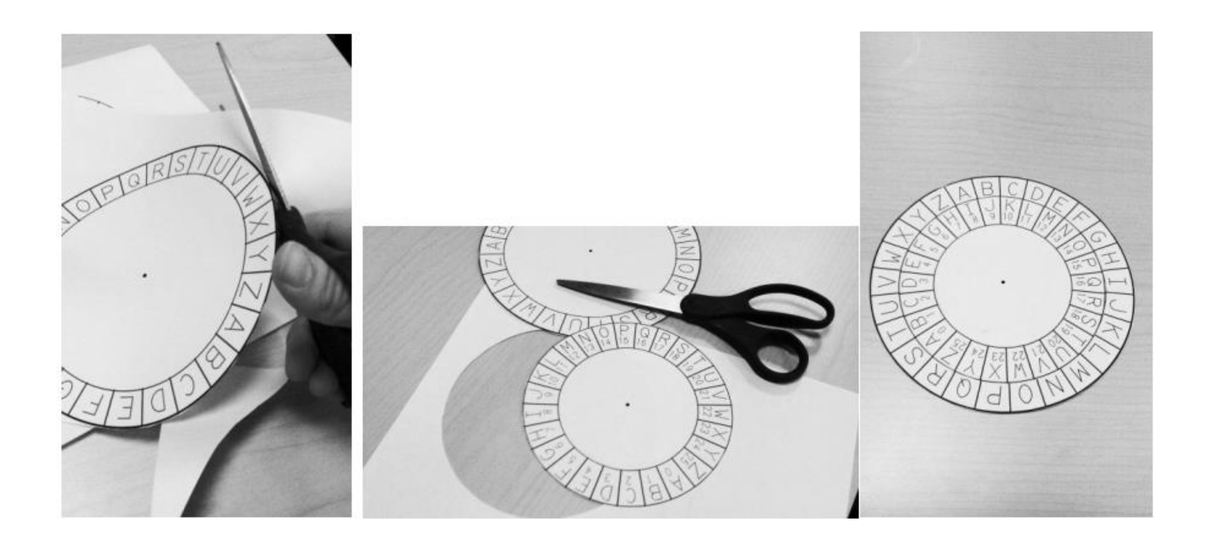
قم بقص الدائرتين وضعهم فوق بعض كما في الشكل التالي:



The inner circle of the cipher wheel cutout.



The outer circle of the cipher wheel cutout.



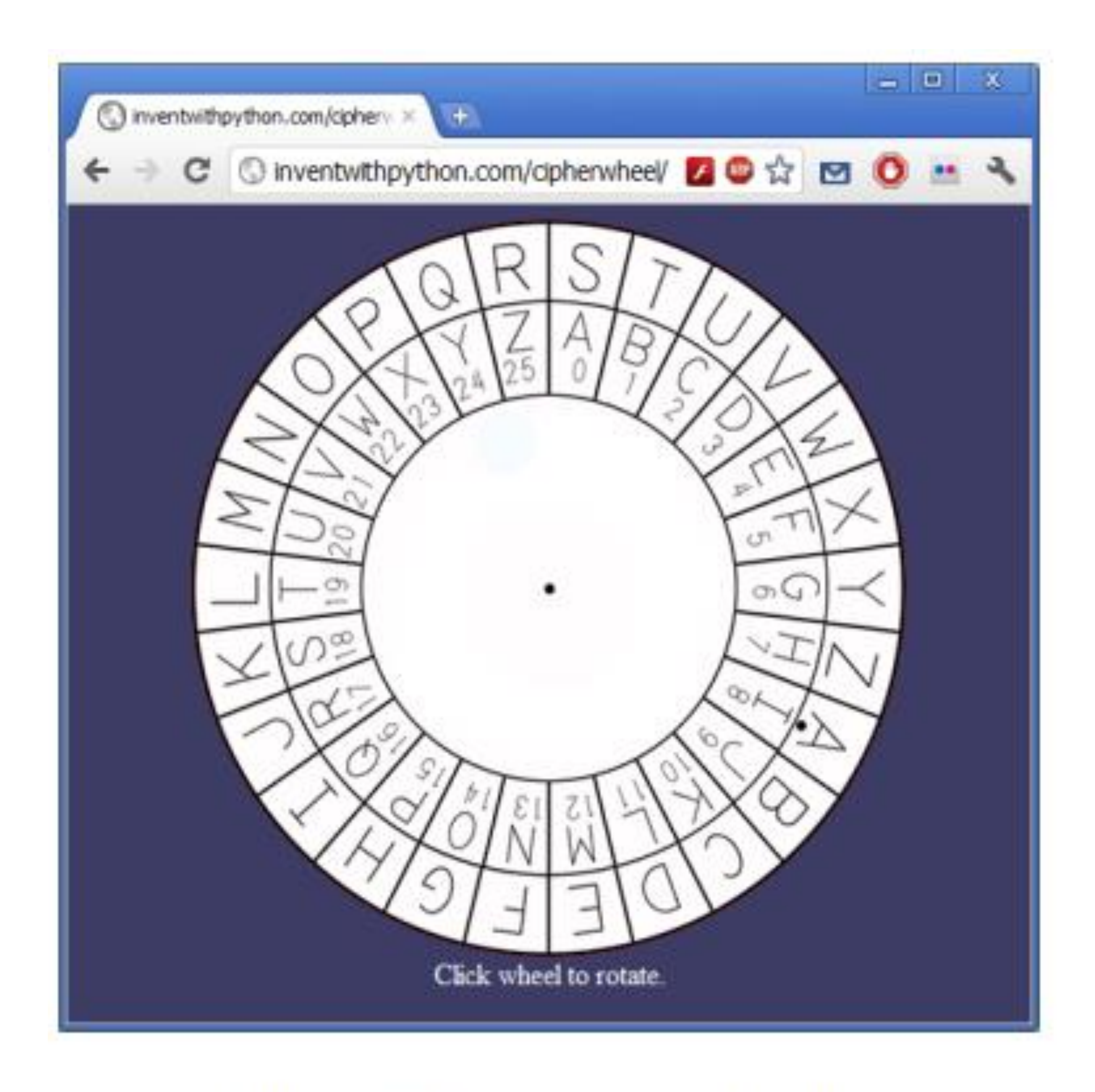
بعد القيام بقص حواف الدائرتين قم بتثبيت الدائرتين فوق بعض من خلال تثبيت دبوس في المنتصف بحيث تصبح قادر على تدويرها.

الآن أنت تملك أداة لخلق رسائل سرية مشفرة بشيفرة قيصر.

يمكنك استخدام دولاب التشفير هذا بشكل أونلاين من الموقع

http://invpy.com/cipherwheel

من أجل تدوير الدولاب اضغط عليه وقم بتحريك الماوس ومن أجل إيقاف دوران الدولاب اضغط عليه مرة ثانية.



The online cipher wheel

كيف تقوم بالتشفير باستخدام دولاب التشفير:

في البداية قم بكتابة الرسالة التي تريد تشفيرها على ورقة وباللغة الإنكليزية

في هذا المثال سوف نقوم بتشفير الرسالة التالية: "The secret password is Rosebud"

ثم قم بتدوير الحلقة الداخلية حتى تصل إلى الأحرف المطابقة مع الحلقة الخارجية

لاحظ أنه في الحلقة الخارجية يوجد نقطة بجانب الحرف A

انظر إلى الرقم المجاور للنقطة في الحلقة الخارجية

هذا الرقم يسمى مفتاح التشفير encryption key

مفتاح التشفير هو المفتاح السري الذي سوف يستخدم في عملية التشفير وعملية فك التشفير

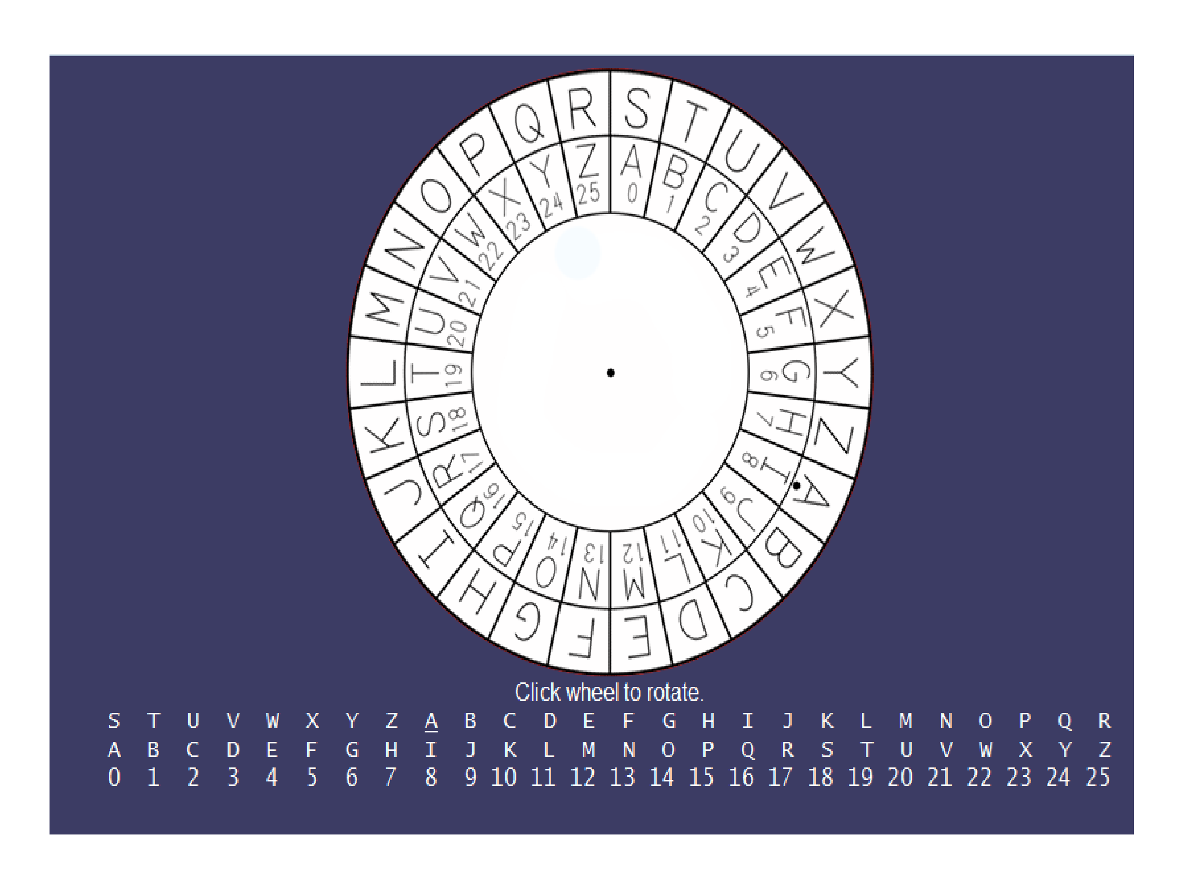
أي شخص لا يملك مفتاح التشفير لا يستطيع قراءة الرسالة المشفرة

في الشكل التالي الحرف A من الحلقة الخارجية هو فوق الرقم 8 من الحلقة الداخلية وهذا يعني أنه سوف يتم استخدام المفتاح 8 لتشفير الرسالة.

شيفرة قيصر تستخدم مفاتيح تشفير من 0 to 25

سوف نستخدم المفتاح 8 في تشفير الرسالة في هذا المثال.

يجب أن تحافظ على مفتاح التشفير بشكل سري، أي شخص يعرف المفتاح السري للتشفير يمكنه قراءة الرسالة المشفرة.



من أجل كل حرف في الرسالة المراد تشفيرها سوف نقوم بالبحث عن هذا الحرف في الحلقة الخارجية واستبداله بالحرف المقابل له في الحلقة الداخلية.

أو حرف في رسالتنا هو T أولاً سوف نقوم بإيجاد هذا الحرف في الحلقة الخارجية ومن ثم الحرف المقابل له في الحلقة الداخلية وهو B

عند استخدام مفتاح تشفير مختلف عن 8 سوف يتغير الحرف المقابل

الحرف الثاني في رسالتنا هو h والذي يقابل الحرف p

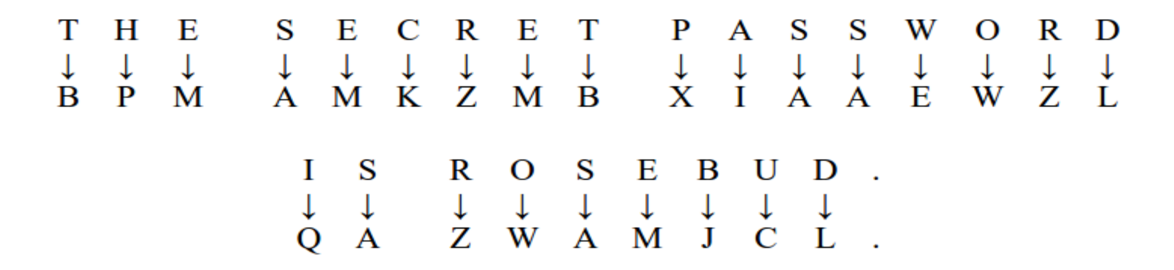
ثم الحرف e والذي يقابل الحرف m

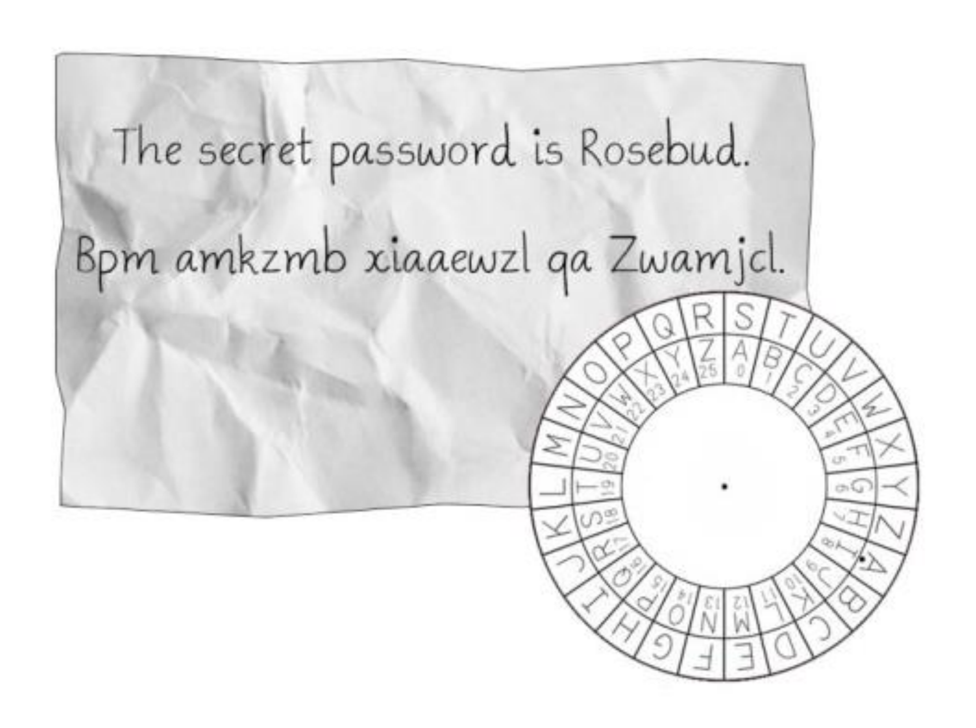
وعندما ننتهي من تشفير كامل الرسالة فإن الرسالة سوف تتحول من

"The secret password is Rosebud"

إلى "Bpm amkzmb xiaaewzl qa Zwamjcl"

الآن يمكنك إرسال الرسالة إلى الشخص الهدف ولن يتمكن أحد من فهم محتوى هذه الرسالة إذا لم تقم بإخباره بقيمة مفتاح التشفير السري (الرقم 8 في هذا المثال)





من أجل توفير الوقت بعد ما قمت بالبحث عن الحرف t واستبداله بالحرف المقابل له b بإمكانك استبدال كل حرف t موجود في الرسالة بالحرف b دون العودة للحلقة مرة ثانية والبحث عن نفس الحرف.

كيف تقوم بفك التشفير باستخدام دولاب التشفير:

من أجل فك تشفير النص المشفر decrypt cipher text

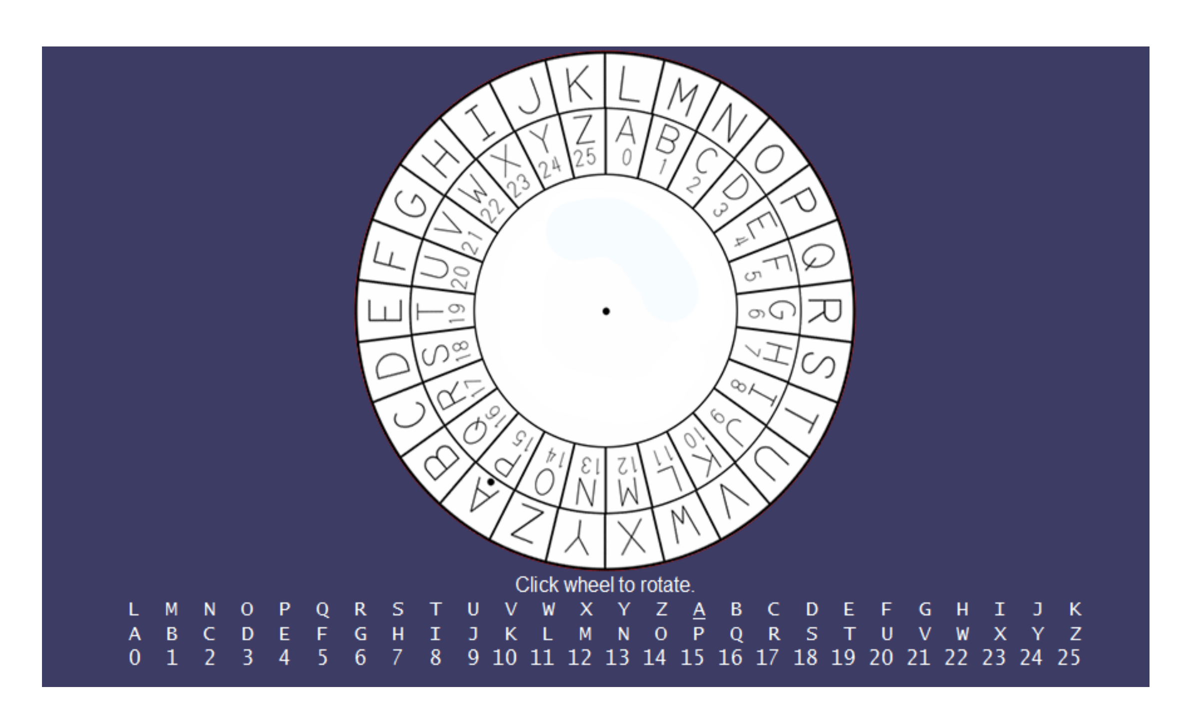
سوف نتبع نفس العملية السابقة ولكن بشكل معاكس، سوف نبحث عن الحرف في الحلقة الداخلية ونقوم باستبداله بالحرف المقابل له من الحلقة الخارجية.

مثلاً لقد تلقينا الرسالة المشفرة التالية “Iwt ctl ephhldgs xh Hldgsuxhw” أنت أو أي شخص أخر لن تكونوا قادرين على فك تشفير هذه الرسالة إذا لم تحصلوا على مفتاح التشفير السري (أو أن تكون هاكر ذكي)

لنفترض أن المفتاح السري الذي استخدم في تشفير هذه الرسالة هو الرقم 15

قم بتدوير الحلقة إلى أن تصبح النقطة في الحلقة الخارجية مقابلة للرقم 15

كما في الشكل التالي:



الحرف الأول في الرسالة المشفرة هو I نقوم بإيجاد هذا الحرف على الحلقة الداخلية واستبداله بالحرف المقابل له في الحلقة الخارجية وهو T

الحرف w من الرسالة المشفرة سوف يتم استبداله بالحرف H وبنفس الطريقة نستبدل كل حرف بالحرف المقابل له وبالتالي نستطيع فك شيفرة الرسالة وتحويلها من نص مشفر cipher text إلى نص صريح plain text وهو "The new password is Swordfish"

I	W	T	C	T	L	E	P	H	H	L	D	G	S
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
T	H	E	N	E	W	P	A	S	S	W	O	R	D

X	H	H	L	D	G	S	U	X	H	W .
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
I	S	S	W	O	R	D	F	I	S	H .

إذا استخدمت مفتاح خاطئ (مثل 16) بدل المفتاح الصحيح (15) فإن نص الرسالة بعد فك التشفير سوف يكون "Sgd mdv ozrrvnqc hr Rvnqcehrg" وهذا النص الصريح لا يبدو أنه صريح وهو غير مفهوم وخاطئ.

إذا لم يتم استخدام المفتاح الصحيح فإنه لن يتم فك تشفير الرسالة بشكل صحيح.

القيام بتشفير قيصر باستخدام الورقة والقلم:

يمكن القيام بتشفير قيصر بدون الحاجة لاستخدام الدولاب وذلك باستخدام الورقة والقلم فقط

قم بكتابة الأحرف من A to Z مع الأرقام من 0 to 25

بحيث يكون الرقم 0 تحت الحرف A والرقم 25 تحت الحرف Z

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**	**J**	**K**	**L**	**M**
0	1	2	3	4	5	6	7	8	9	10	11	12
N	**O**	**P**	**Q**	**R**	**S**	**T**	**U**	**V**	**W**	**X**	**Y**	**Z**
13	14	15	16	17	18	19	20	21	22	23	24	25

من أجل التشفير نقوم بإيجاد الرقم الموجود تحت الحرف المراد تشفيره ثم نقوم بجمع قيمة مفتاح التشفير مع الرقم

الرقم الناتج من عملية الجمع سوف يكون تحت الحرف المشفر.

مثال:

نريد تشفير الرسالة "Hello How are you?" باستخدام مفتاح التشفير 13

أولاً نقوم بإيجاد الرقم تحت الحرف H والذي هو الرقم 7

ثم نقوم بجمع هذا الرقم مع قيمة مفتاح التشفير 7+13=20

الرقم 20 موجود تحت الحرف U

إذا الحرف U هو الحرف المشفر للحرف H

من أجل تشفير الحرف E

نقوم بجمع الرقم الموجود تحت الحرف E وهو الرقم 4 مع قيمة مفتاح التشفير 13

4+13=17

الرقم 17 يقع تحت الحرف R

إذا الحرف R هو تشفير للحرف E وهكذا

هذه الطريقة تعمل بشكل جيد إلى أن نصل إلى الحرف O

الرقم تحت الحرف O هو 14

ولكن عندما نقوم بجمع هذا الرقم مع قيمة مفتاح التشفير 14+13=27

ولكن الأرقام التي لدينا هي فقط حتى 25

إذا كان ناتج مجموع الرقم تحت الحرف مع قيمة مفتاح التشفير أكبر من 26 فيجب أن نطرح منه 26

أي 27-26=1 والحرف الموجود فوق الرقم 1 هو B

إذا الحرف B هو الحرف المشفر للحرف O وذلك عند استخدام مفتاح التشفير 13

باستخدام نفس الطريقة على كل أحرف الرسالة "Hello How are you?" نحصل على الرسالة المشفرة "Uryyb Ubj ner Ibh?"

خطوات تشفير الرسالة هي:

١- اختيار مفتاح سري للتشفير من 1 to 25
٢- إيجاد الرقم تحت الحرف بالنص الصريح
٣- جمع قيمة هذا الرقم مع قيمة مفتاح التشفير
٤- إذا كان الرقم الناتج عن عملية الجمع أكبر من 26 نقوم بطرح 26 منه
٥- إيجاد الحرف الموجود فوق الرقم الناتج، هذا الحرف هو الحرف المشفر
٦- تكرار الخطوات من 2 to 5 من أجل كل حرف في الرسالة

الجدول التالي يظهر كيف تتم هذه العملية من أجل تشفير الرسالة "Hello How are you?" باستخدام مفتاح التشفير 13

Plaintext Letter	Plaintext Number	+	Key	Result	Subtract 26?	Result	Ciphertext Letter
H	7	+	13	= 20		= 20	20 = U
E	4	+	13	= 17		= 17	17 = R
L	11	+	13	= 24		= 24	24 = Y
L	11	+	13	= 24		= 24	24 = Y
O	14	+	13	= 27	- 26	= 1	1 = B
H	7	+	13	= 20		= 20	20 = U
O	14	+	13	= 27	- 26	= 1	1 = B
W	22	+	13	= 35	- 26	= 9	9 = J
A	0	+	13	= 13		= 13	13 = N
R	17	+	13	= 30	- 26	= 4	4 = E
E	4	+	13	= 17		= 17	17 = R
Y	24	+	13	= 37	- 26	= 11	11 = L
O	14	+	13	= 27	- 26	= 1	1 = B
U	20	+	13	= 33	- 26	= 7	7 = H

من أجل عملية فك التشفير decrypt نقوم بعملية طرح قيمة مفتاح التشفير بدل من عملية الجمع.

من أجل الحرف المشفر B

الرقم الموجود تحت هذا الحرف هو 1

نقوم بطرح قيمة مفتاح التشفير من هذا الرقم 12-=1-13

عندما يكون الناتج أقل من 0 (عدد سالب) نقوم بإضافة العدد 26

-12 +26=14

الحرف الموجود فوق الرقم 14 هو O

وبالتالي عند فك تشفير الحرف B نحصل على الحرف O

الجدول التالي يظهر خطوات عملية فك التشفير:

Ciphertext Letter	Ciphertext Number	-	Key	Result	Add 26?	Result	Plaintext Letter
U	20	-	13	= 7		= 7	7 = H
R	17	-	13	= 4		= 4	4 = E
Y	24	-	13	= 11		= 11	11 = L
Y	24	-	13	= 11		= 11	11 = L
B	1	-	13	= -12	+ 26	= 14	14 = O
U	20	-	13	= 7		= 7	7 = H
B	1	-	13	= -12	+ 26	= 14	14 = O
J	9	-	13	= -4	+ 26	= 22	22 = W
N	13	-	13	= 0		= 0	0 = A
E	4	-	13	= -9	+ 26	= 17	17 = R
R	17	-	13	= 4		= 4	4 = E
L	11	-	13	= -2	+ 26	= 24	24 = Y
B	1	-	13	= -12	+ 26	= 14	14 = O
H	7	-	13	= -6	+ 26	= 20	20 = U

التشفير مزدوج القوة:

يمكن أن تعتقد أن تشفير الرسالة مرتين باستخدام مفتاحي تشفير مختلفين سوف يضاعف قوة التشفير ولكن هذا ليس صحيح في تشفير قيصر (ومعظم الشيفرات الأخرى)

سوف نقوم بتجربة التشفير المزدوج لنعرف السبب

إذا قمنا بتشفير الكلمة "KITTEN" باستخدام مفتاح التشفير 3 فإن النتيجة ستكون الكلمة المشفرة "NLWWHQ"

إذا قمنا بتشفير الكلمة "NLWWHQ" باستخدام المفتاح 4 فإن النتيجة ستكون الكلمة المشفرة "RPAALU" ولكن هذه النتيجة هي نفس النتيجة التي سوف نحصل عليها في حال قمنا بتشفير الكلمة "KITTEN" باستخدام المفتاح 7

عملية التشفير المزدوج هي عملية تشفير عادية وليست عملية قوية

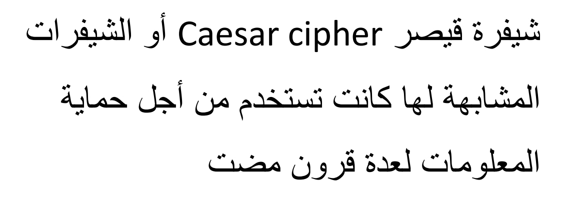
السبب هو عندما نقوم بالتشفير باستخدام المفتاح 3 فإننا نقوم بإضافة 3 إلى أرقام أحرف النص الصريح ثم عندما نقوم بعملية التشفير باستخدام المفتاح 4 فإننا نقوم بإضافة الرقم 4 إلى أرقام أحرف النص الصريح، ولكن إضافة 3 ثم 4 يعطي نفس النتيجة في حال إضافة 7

التشفير المضاعف باستخدام المفتاحين 3 and 4 هو نفسه التشفير العادي باستخدام المفتاح 7

في معظم الشيفرات فإن عملية التشفير لأكثر من مرة لا تؤمن قوة إضافية للشيفرة

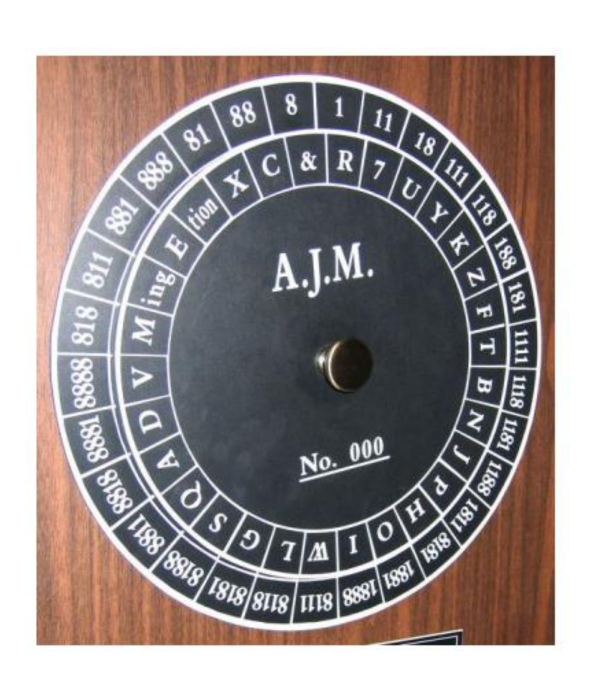
في الحقيقة إذا قمت بتشفير النص الصريح plain text باستخدام مفتاحين مجموعهما يساوي 26 فإن النص المشفر cipher text سوف يكون نفس النص الصريح الأصلي.

استخدام البرمجة للقيام بالتشفير:



شيفرة قيصر Caesar cipher أو الشيفرات المشابهة لها كانت تستخدم من أجل حماية المعلومات لعدة قرون مضت

الشكل التالي هو لقرص تشفير صمم من قبل Albert Myer والذي استخدم في الحرب الأهلية الأميركية عام 1863



إذا كان لديك رسالة طويلة جداً وتريد تشفيرها (كتاب مثلاً) فسوف تحتاج لأيام أو حتى أسابيع من أجل إتمام عملية التشفير

البرمجة يمكن أن تساعدك للقيام بهذه المهمة

الكمبيوتر يستطيع القيام بهذه المهمة ويستطيع تشفير كمية كبيرة من النصوص والكلمات خلال أجزاء من الثانية ولكن نحن بحاجة لأن نتعلم كيفية إرشاد الكمبيوتر للقيام بهذه المهمة

يجب أن نتحدث مع الكمبيوتر بلغة يستطيع أن يفهمها، لحسن الحظ فإن تعلم لغة البرمجة هو أسهل بكثير من تعلم أي لغة أجنبية أخرى كاليابانية أو الاسبانية.

كل ما عليك معرفته هو أساسيات الرياضيات بالإضافة إلى تنزيل برنامج مجاني python والذي سوف نقوم بشرحه في الفصل التالي.



تنصيب البايثون

محتوى هذا الفصل:

- تنزيل وتنصيب البايثون
- تنزيل Pyperclip module
- كيف تبدأ مع IDLE
- التنسيق المستخدم في هذا الكتاب
- نسخ ولصق النصوص

"Privacy in an open society also requires cryptography. If I say something, I want it heard only by those for whom I intend it. If the content of my speech is available to the world, I have no privacy."

Eric Hughes, "A Cypherpunk's Manifesto", 1993

تنزيل وتنصيب بايثون:

قبل أن تستطيع البدء بالبرمجة يجب أن تقوم بتنصيب برنامج مفسر (مترجم) أوامر لغة البرمجة بايثون والذي يسمى Python interpreter

المترجم هو البرنامج الذي يقوم بفهم التعليمات التي تقوم بكتابتها بلغة البايثون وبدون هذا المترجم فإن جهازك الكمبيوتر لن يستطيع فهم التعليمات

من الآن ولاحقاً سوف نشير إلى مترجم بايثون python interpreter باسم بايثون فقط

لأننا نريد أن نكتب برامج بلغة البايثون فنحن نحتاج لتنزيل البايثون من موقعه الرسمي

http://www.python.org

عملية التنصيب تختلف قليلاً بحسب نوع نظام التشغيل الذي تستخدمه فيما إذا كان نظام ويندوز أو نظام أبونتو لينكس أو أنظمة تشغيل أخرى

يمكنك مشاهدة مقاطع فيديو لكيفية تنصيب البايثون من خلال الموقع التالي

http://invpy.com/installing



ملاحظة هامة: تأكد من تنصيب python3 وليس python2 في هذا الكتاب سوف نستخدم python3 وسوف نحصل على أخطاء في حال تطبيق برامج هذا الكتاب على python2

تنصيب بايثون على نظام ويندوز:

يوجد قائمة من الروابط على الجانب الأيسر في الصفحة الرئيسية للموقع http://www.python.org اضغط على Download link لتذهب إلى صفحة التحميل ثم أبحث عن الملف Python 3.4.3 Windows Installer واضغط على الرابط الخاص به من

أجل تحميل البايثون الخاص بنظام الويندوز (إذا كان يوجد أحدث من الإصدار python 3.4.3 قم بتحميله)

بعد التحميل قم بفتح الملف المحمل python-3.4.3.msi للبدء بعملية تنصيب البايثون (إذا لم يعمل قم بالضغط عليه بالزر اليميني واختر تنصيب)

عندما يبدأ برنامج التنصيب بالعمل أضغط على Next وعند انتهاء عملية التنصيب أضغط على Finish

تنصيب بايثون على نظام أبونتو لينكس:

قم بفتح التيرمينل وادخل الأمر التالي:

```
sudo apt-get install python3.3
```

وبعد الضغط على enter ستكون بحاجة لإدخال root password

كما أنك ستكون بحاجة لتنصيب IDLE software باستخدام الأمر التالي:

```
Sudo apt-get install idle3
```

وأيضاً سوف يطلب منك إدخال root password

تنزيل pyperclip.py:

تقريباً كل البرامج في هذا الكتاب تستخدم الوحدة pyperclip.py module هذا الموديول يحوي على توابع functions تسمح لبرنامجك بنسخ ولصق النص

هذا الموديول لا يكون مع بايثون بشكل افتراضي ولكن بإمكانك الحصول عليه من

https://inventwithpython.com/pyperclip.py

هذا الملف يجب أن يكون في نفس المجلد الذي يحوي على برنامج البياثون الذي تقوم بكاتبته وإلا سوف تظهر لك رسالة الخطأ التالية عن تشغيل البرنامج

```
ImportError: No module named pyperclip
```

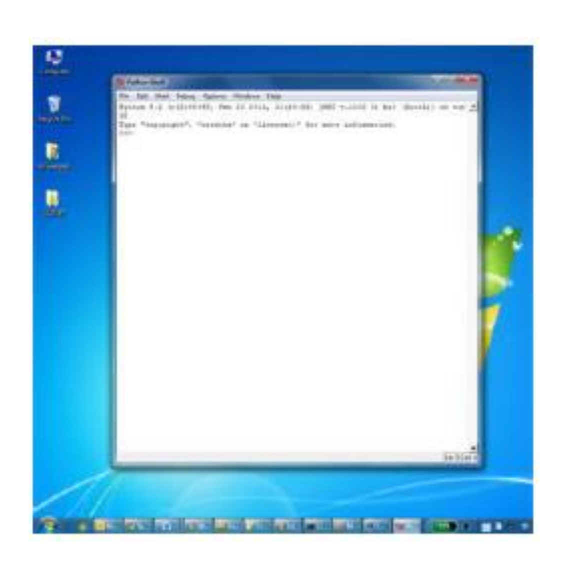
تشغيل IDLE:

IDLE: Interactive DeveLopment Environment بيئة التطوير التفاعلية.

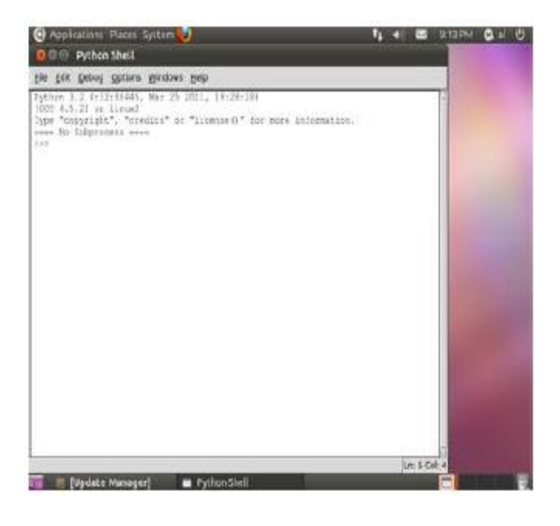
سوف نقوم باستخدام IDLE software من أجل كتابة برامجنا وتشغيلها.

مترجم بايثون هو البرنامج الذي يقوم بترجمة وتنفيذ البرامج المكتوبة بلغة البايثون بينما IDLE هو البرنامج الذي سوف تقوم بكتابة تعليمات البرنامج في داخله

إذا كنت تستخدم نظام Windows7 افتح قائمة ابدأ واكتب "IDLE" في شريط البحث ثم قم بتشغيل IDLE والذي يعتبر الواجهة الرسومية للبايثون Python GUI



إذا كنت تستخدم نظام أبونتو لينكس قم بكتابة idle3 في التيرمينل



هذه النافذة التي تظهر عند تشغيل بيئة التطوير التفاعلية IDLE تسمى الشيل التفاعلية interactive shell:

```
Python 3.4.3 (v3.4.3:9b73f1c3e601, Feb 24 2015, 22:43:06) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 
```

الشيل shell هو برنامج يسمح لك بكتابة التعليمات في جهاز الكمبيوتر.

بايثون شيل Python shell يسمح لك بكتابة تعليمات البياثون في داخله ثم يقوم بإرسال هذه التعليمات إلى مترجم أو مفسر أوامر البايثون Python interpreter والذي يقوم بتشغيل أو تنفيذ هذه التعليمات.

تستطيع كتابة تعليمات أو أوامر بايثون في الشيل ولأن هذه الشل تفاعلية فإن جهاز الكمبيوتر سوف يقرأ هذه التعليمات ويقوم بتنفيذها بشكل فوري.

ميزات البرامج:

كتاب "فك الشيفرات السرية باستخدام البايثون" مختلف عن كتب البرمجة الأخرى لأنه يركز على الكود البرمجي source code للبرامج الكاملة.

بدل من أن يعلمك مبادئ البرمجة فهو يساعدك على كتابة برامجك الخاصة.

هذا الكتاب يظهر لك برامج كاملة ثم يتم شرح كيف تعمل هذه البرامج.

عندما تقرأ هذا الكتاب قم بكتابة الكود البرمجي للبرامج المشرحة في هذا الكتاب بنفس الترتيب أو يمكنك تحميل ملفات الكود البرمجي للبرامج هذا الكتاب من الموقع:

http://invpy.com/hackingsource

البايثون هي لغة برمجة مقروءة وسهلة ومن خلال قراءة الفصول الأولى من هذا الكتاب يمكنك فهم العمل الذي سوف يقوم الكود بتنفيذه، إذا كنت قد انتقلت مباشراً لقراءة الفصول القادمة وشعرت بالضياع حاول العودة للفصول السابقة

أرقام الأسطر والمسافات:

عندما تقوم بإدخال الكود البرمجي بنفسك لا تقم بكتابة أرقام الأسطر التي تظهر في بداية كل سطر

مثلاً إذا وجدت الكود التالي:

```
1. number = random.randint(1, 20)
2. spam = 42
3. print('Hello world!')
```

فلا تقم بكتابة الأرقام الموجود في بداية كل سطر ولا المسافات الموجودة بعد هذه الأرقام

فقط قم بكتابة الأوامر كما في الشكل التالي:

```
number = random.randint(1, 20)
spam = 42
print('Hello world!')
```

هذه الأرقام فقط تستخدم في الشرح من أجل الإشارة إلى أسطر معينة في الكود وهي ليست جزء من الكود البرمجي الفعلي.

بالإضافة لذلك تأكد من إدخال الكود البرمجي بشكل مطابق تماماً للكود الموجود في هذا الكتاب وهذا يتضمن حالة الأحرف كبيرة أو صغيرة لأن البايثون هي لغة حساسة لحالة الأحرف

مثلاً في بايثون HELLO تختلف عن hello

لاحظ أن بعض الأسطر لا تبدأ من حافة الجانب الأيسر للصفحة ولكنها تبدأ بعد مسافة معينة 4 or 8 فراغات، كن متأكد من ادخال الرقم الصحيحة للمسافة في بداية كل سطر(كل الأحرف في IDLE لها نفس العرض، يمكنك معرفة عدد المسافات من خلال معرفة عدد الأحرف في السطر الأعلى من السطر الذي تنظر إليه)

مثلاً يمكنك أن ترى أن السطر الثاني متقدم بمسافة أربع خطوات لأن أحرف (Whil) في السطر الأعلى هي أربعة أحرف

السطر الثالث متقدم بمسافة أربع خطوات عن السطر الثاني لأن (if n) هي أربع خطوات

```
while spam < 10:
    if number == 42:
        print('Hello')
```

إلتفاف النص:

بعض الأسطر في الكود طويلة جداً ولا تتسع في سطر واحد فإن النص سوف يلف إلى السطر الثاني، عندما تقوم بكتابة هذه الأسطر في ملف نصي قم بإدخال النص بسطر واحد بدون أن تقوم بالنزول إلى السطر الثاني

يمكنك معرفة متى يبدأ السطر من خلا النظر إلى رقم السطر في الجانب الأيسر

كما يظهر في المثال التالي الذي يحوي على سطرين فقط

```
1. print('This is the first line! xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx
xxxxxxxxxxxxxx')
2. print('This is the second line!')
```

تخطيط البرنامج بشكل أون لاين:

تخطيط البرنامج أو انشاء رسم تصوري له يظهر الكود في نفس الطريقة التي سيقوم الكمبيوتر بتنفذه

بمكنك زيارة الموقع التالي http://invpy.com/hackingtraces لرؤية مخطط كل برنامج في هذا الكتاب.

هذا الموقع يحوي على ملاحظات في كل خطوة وهي تشرح العمل الذي يقوم البرنامج بتنفيذه وهذا يساعدك على فهم البرامج بشكل أفضل.

فحص الكود باستخدام أداة Diff:

طريقة سهلة لتعلم البايثون هي كتابة الكود البرمجي لهذه البرامج ولكن يمكن أن تكتب إحدى التعليمات بشكل خاطئ سهواً وهذا يؤدي إلى خطأ في تنفيذ البرنامج

يمكنك نسخ النص البرمجي الذي قمت بكتابته ولصقه في online diff tool الموجودة في موقع الكتاب، هذه الأداة تظهر أي اختلاف بين الكود البرمجي في الكتاب والكود البرمجي الذي قمت أنت بكتابته

وهذه الطريقة تسهل عملية إيجاد أي خطأ كتابي في برنامجك

هذه الأداة موجودة في http://invpy.com/hackingdiff

فديو تعليمي يشرح طريقة استخدام هذه الأداة موجود في http://invpy.com/hackingvideos

نسخ ولصق النص:

عملية نسخ ولص النص مفيداً جداً وخاصة لأن العديد من النصوص التي سيتم تشفيرها و فك تشفيرها هي نصوص طويلة وبدلاً من كتابتها يمكنك نسخ النص من هذا الكتاب أو من النسخة الالكترونية الأصلية لهذا الكتاب ولصقه في IDLE

يمكن رؤية كيفية القيام بهذه العملية من خلال http://invpy.com/copypaste

روابط معلومات أخرى:

يوجد العديد من المصادر التي يمكن أن تساعدك في تعلم التشفير والبرمجة ولكنك لن تكون بحاجة لتعلم كل الأمور الآن ولكن في بعض الأحيان أنت بحاجة لمعرفة هذه التفاصيل والشروحات الإضافية، لذلك تم إضافة رابط معلومات إضافية للمساعدة على فهم بعض محتويات هذا الكتاب

رغم أن هذا الكتاب لا يحوي معلومات وشروحات خطيرة رجاءً لا تستخدم محتوى هذا الكتاب بشكل سيء

البرمجة والتشفير:

البرمجة والتشفير مهارتان منفصلتان ولكن تعلمها معاً هو أمر مفيد لأن الكمبيوتر يستطيع القيام بالتشفير بشكل أسرع بكثير من الإنسان

مثلاً: التالي هو النص الكامل لرواية وليم شكسبير "روميو وجوليت" وهو مشفر بشيفرة استبدال بسيطة

WUMIU FOT LZNCIB JWUNURZI Dguwzh Bku guzhigunth, qubg fncei co tcrocbx, Co afcw Piwuof, kgiwi ki nfx uzw hdioi, Awum fodciob rwztri qwife bu oik mzbcox, Kgiwi dcpcn qnuut mfeih dcpcn gfoth zodnifo. Awum auwbg bgi afbfn nucoh ua bgihi bku auih F jfcw ua hbfw-dwuhh't nupiwh bfei bgicw ncai; Kguni mchftpiobzwit jcbiuzh upiwbgwukh Tu kcbg bgicw tifbg qzwx bgicw jfwiobh' hbwcai. Bgi aifwazn jfhhfri ua bgicw tifbg-mfwe't nupi, Fot bgi duobcozfodi ua bgicw jfwiobh' wfri, Kgcdg, qzb bgicw dgcntwio'h iot, ouzrgb duznt wimupi, Ch ouk bgi bku guzwh' bwfaacd ua uzw hbfri; Bgi kgcdg ca xuz kcbg jfbciob ifwh fbbiot, Kgfb giwi hgfnn mchh, uzw bucn hgfnn hbwcpi bu miot. WUMIU FOT LZNCIB FDB C HDIOI C Piwuof. F jzqncd jnfdi. [Iobiw HFMJHUO fot RWIRUWX, ua bgi guzhi ua Dfjznib, fwmit kcbg hkuwth fot qzdeniwh] HFMJHUO Rwiruwx, u' mx kuwt, ki'nn oub dfwwx dufnh. RWIRUWX Ou, auw bgio ki hguznt qi dunnciwh. HFMJHUO C mifo, fo ki qi co dguniw, ki'nn twfk. RWIRUWX Fx, kgcni xuz ncpi, twfk xuzw oide uzb u' bgi dunnfw. HFMJHUO C hbwcei yzcdenx, qicor mupit. RWIRUWX Qzb bguz fwb oub yzcdenx mupit bu hbwcei. HFMJHUO F tur ua bgi guzhi ua Muobfrzi mupih mi. RWIRUWX Bu mupi ch bu hbcw; fot bu qi pfncfob ch bu hbfot: bgiwiauwi, ca bguz fwb mupit, bguz wzoo'hb fkfx. HFMJHUO F tur ua bgfb guzhi hgfnn mupi mi bu hbfot: C kcnn bfei bgi kfnn ua fox mfo uw mfct ua Muobfrzi'h. RWIRUWX Bgfb hgukh bgii f kife hnfpi; auw bgi kifeihb ruih bu bgi kfnn. HFMJHUO Bwzi; fot bgiwiauwi kumio, qicor bgi kifeiw pihhinh, fwi ipiw bgwzhb bu bgi kfnn: bgiwiauwi C kcnn jzhg Muobfrzi'h mio awum bgi kfnn, fot bgwzhb gch mfcth bu bgi kfnn. RWIRUWX Bgi yzfwwin ch qibkiio uzw mfhbiwh fot zh bgicw mio. HFMJHUO 'Bch fnn uoi, C kcnn hguk mxhina f bxwfob: kgio C gfpi auzrgb kcbg bgi mio, C kcnn qi dwzin kcbg bgi mfcth, fot dzb uaa bgicw gifth. RWIRUWX Bgi gifth ua bgi mfcth? HFMJHUO Fx, bgi gifth ua bgi mfcth, uw bgicw mfctiogifth; bfei cb co kgfb hiohi bguz kcnb. RWIRUWX Bgix mzhb bfei cb co hiohi bgfb aiin cb. HFMJHUO Mi bgix hgfnn aiin kgcni C fm fqni bu hbfot: fot 'bch eouko C fm f jwibbx jcidi ua anihg. RWIRUWX 'Bch kinn bguz fwb oub achg; ca bguz gfthb, bguz gfthb qiio juuw Lugo. Twfk bgx buun! giwi dumih bku ua bgi guzhi ua bgi Muobfrzih. HFMJHUO Mx ofeit kifjuo ch uzb: yzfwwin, C kcnn qfde bgii. RWIRUWX Guk! bzwo bgx qfde fot wzo? HFMJHUO Aifw mi oub. RWIRUWX Ou, mfwwx; C aifw bgii! HFMJHUO Nib zh bfei bgi nfk ua uzw hctih; nib bgim qirco. RWIRUWX C kcnn awuko fh C jfhh qx, fot nib bgim bfei cb fh bgix nchb. HFMJHUO Ofx, fh bgix tfwi. C kcnn qcbi mx bgzmq fb bgim; kgcdg ch f tchrwfdi bu bgim, ca bgix qifw cb. [Iobiw FQWFGFM fot QFNBGFHFW] FQWFGFM Tu xuz qcbi xuzw bgzmq fb zh, hcw? HFMJHUO C tu qcbi mx bgzmq, hcw. FQWFGFM Tu xuz qcbi xuzw bgzmq fb zh, hcw? HFMJHUO [Fhcti bu RWIRUWX] Ch bgi nfk ua uzw hcti, ca C hfx fx? RWIRUWX Ou. HFMJHUO Ou, hcw, C tu oub qcbi mx bgzmq fb xuz, hcw, qzb C qcbi mx bgzmq, hcw. RWIRUWX Tu xuz yzfwwin, hcw? FQWFGFM Yzfwwin hcw! ou, hcw. HFMJHUO Ca xuz tu, hcw, C fm auw xuz: C hiwpi fh ruut f mfo fh xuz. FQWFGFM Ou qibbiw. HFMJHUO Kinn, hcw. RWIRUWX Hfx 'qibbiw:' giwi dumih uoi ua mx mfhbiw'h ecohmio. HFMJHUO Xih, qibbiw, hcw. FQWFGFM Xuz nci. HFMJHUO Twfk, ca xuz qi mio. Rwiruwx, wimimqiw bgx hkfhgcor qnuk. [Bgix acrgb] [Iobiw QIOPUNCU] QIOPUNCU Jfwb, auunh! Jzb zj xuzw hkuwth; xuz eouk oub kgfb xuz tu. [Qifbh tuko bgicw hkuwth] [Iobiw BXQFNB] BXQFNB Kgfb, fwb bguz twfko fmuor bgihi gifwbnihh gcoth? Bzwo bgii, Qiopuncu, nuue zjuo bgx tifbg. QIOPUNCU C tu qzb eiij bgi jifdi: jzb zj bgx hkuwt, Uw mfofri cb bu jfwb bgihi mio kcbg mi. BXQFNB Kgfb, twfko, fot bfne ua jifdi! C gfbi bgi kuwt, Fh C gfbi ginn, fnn Muobfrzih, fot bgii: Gfpi fb bgii, dukfwt! [Bgix acrgb] [Iobiw, hipiwfn ua qubg guzhih, kgu luco bgi awfx; bgio iobiw Dcbcvioh, kcbg dnzqh] Acwhb Dcbcvio Dnzqh, qcnnh, fot jfwbchfoh! hbwcei! qifb bgim tuko! Tuko kcbg bgi Dfjznibh! tuko kcbg bgi Muobfrzih! [Iobiw DFJZNIB co gch ruko, fot NFTX DFJZNIB] DFJZNIB Kgfb ouchi ch bgch? Rcpi mi mx nuor hkuwt, gu! NFTX DFJZNIB F dwzbdg, f dwzbdg! kgx dfnn xuz auw f hkuwt? DFJZNIB Mx hkuwt, C hfx! Unt Muobfrzi ch dumi, Fot anuzwchgih gch qnfti co hjcbi ua mi. [Iobiw MUOBFRZI fot NFTX MUOBFRZI] MUOBFRZI Bguz pcnnfco Dfjznib,--Gunt mi oub, nib mi ru. NFTX MUOBFRZI Bguz hgfnb oub hbcw f auub bu hiie f aui. [Iobiw JWCODI, kcbg Fbbiotfobh] JWCODI Wiqinncuzh hzqlidbh, ioimcih bu jifdi, Jwuafoiwh ua bgch oicrgquzw-hbfcoit hbiin,-- Kcnn bgix oub gifw? Kgfb, gu! xuz mio, xuz qifhbh, Bgfb yziodg bgi acwi ua xuzw jiwocdcuzh wfri Kcbg jzwjni auzobfcoh chhzcor awum xuzw picoh, Uo jfco ua buwbzwi, awum bguhi qnuutx gfoth Bgwuk xuzw mchbimjiw't kifjuoh bu bgi rwzuot, Fot gifw bgi hiobiodi ua xuzw mupit jwcodi. Bgwii dcpcn qwfknh, qwit ua fo fcwx kuwt, Qx bgii, unt Dfjznib, fot Muobfrzi, Gfpi bgwcdi tchbzwq't bgi yzcib ua uzw hbwiibh, Fot mfti Piwuof'h fodciob dcbcvioh Dfhb qx bgicw rwfpi qihiimcor uwofmiobh, Bu kcint unt jfwbchfoh, co gfoth fh unt, Dfoeiw't kcbg jifdi, bu jfwb xuzw dfoeiw't gfbi: Ca ipiw xuz tchbzwq uzw hbwiibh frfco, Xuzw ncpih hgfnn jfx bgi auwaicb ua bgi jifdi. Auw bgch bcmi, fnn bgi wihb tijfwb fkfx: Xuz Dfjznib; hgfnn ru fnuor kcbg mi: Fot, Muobfrzi, dumi xuz bgch fabiwouuo, Bu euok uzw azwbgiw jnifhzwi co bgch dfhi, Bu unt Awii-buko, uzw dummuo lztrmiob-jnfdi. Uodi muwi, uo jfco ua tifbg, fnn mio tijfwb. [Isizob fnn qzb MUOBFRZI, NFTX MUOBFRZI, fot QIOPUNCU] MUOBFRZI Kgu hib bgch fodciob yzfwwin oik fqwufdg? Hjife, oijgik, kiwi xuz qx kgio cb qirfo? QIOPUNCU Giwi kiwi bgi hiwpfobh ua xuzw ftpiwhfwx, Fot xuzwh, dnuhi acrgbcor iwi C tct fjjwufdg: C twik bu jfwb bgim: co bgi cohbfob dfmi Bgi aciwx Bxqfnb, kcbg gch hkuwt jwijfwit, Kgcdg, fh gi qwifbgit tiacfodi bu mx ifwh, Gi hkzor fquzb gch gift fot dzb bgi kcoth, Kgu oubgcor gzwb kcbgfn gchh't gcm co hduwo: Kgcni ki kiwi cobiwdgforcor bgwzhbh fot qnukh, Dfmi muwi fot muwi fot auzrgb uo jfwb fot jfwb, Bcnn bgi jwcodi dfmi, kgu jfwbit icbgiw jfwb. NFTX MUOBFRZI U, kgiwi ch Wumiu? hfk xuz gcm bu-tfx? Wcrgb rnft C fm gi kfh oub fb bgch awfx. QIOPUNCU Mftfm, fo guzw qiauwi bgi kuwhgcjj't hzo Jiiw't auwbg bgi runtio kcotuk ua bgi ifhb, F bwuzqnit mcot twfpi mi bu kfne fqwuft; Kgiwi, zotiwoifbg bgi rwupi ua hxdfmuwi Bgfb kihbkfwt wuubibg awum bgi dcbx'h hcti, Hu ifwnx kfnecor tct C hii xuzw huo: Bukfwth gcm C mfti, qzb gi kfh kfwi ua mi Fot hbuni cobu bgi dupiwb ua bgi kuut: C, mifhzwcor gch faaidbcuoh qx mx uko, Bgfb muhb fwi qzhcit kgio bgix'wi muhb fnuoi, Jzwhzit mx gzmuzw oub jzwhzcor gch, Fot rnftnx hgzoo't kgu rnftnx anit awum mi. MUOBFRZI Mfox f muwocor gfbg gi bgiwi qiio hiio, Kcbg bifwh fzrmiobcor bgi awihg muwocor tik. Fttcor bu dnuzth muwi dnuzth kcbg gch tiij hcrgh; Qzb fnn hu huuo fh bgi fnn-dgiiwcor hzo Hguznt co bgi azwbgihb ifhb qirco bu twfk Bgi hgftx dzwbfcoh awum Fzwuwf'h qit, Fkfx awum bgi ncrgb hbifnh gumi mx gifpx huo, Fot jwcpfbi co gch dgfmqiw jioh gcmhina, Hgzbh zj gch kcotukh, nudeh afw tfxncrgb uzb Fot mfeih gcmhina fo fwbcacdcfn ocrgb: Qnfde fot juwbiobuzh mzhb bgch gzmuzw jwupi, Zonihh ruut duzohin mfx bgi dfzhi wimupi. QIOPUNCU Mx ouqni zodni, tu xuz eouk bgi dfzhi? MUOBFRZI C oicbgiw eouk cb ouw dfo nifwo ua gcm. QIOPUNCU Gfpi xuz cmjuwbzoit gcm qx fox mifoh? MUOBFRZI Qubg qx mxhina fot mfox ubgiw awcioth: Qzb gi, gch uko faaidbcuoh' duzohinnuw, Ch bu gcmhina--C kcnn oub hfx guk bwzi-- Qzb bu gcmhina hu hidwib fot hu dnuhi, Hu afw awum huzotcor fot tchdupiwx, Fh ch bgi qzt qcb kcbg fo iopcuzh kuwm, Iwi gi dfo hjwift gch hkiib nifpih bu bgi fcw, Uw titcdfbi gch qifzbx bu bgi hzo. Duznt ki qzb nifwo awum kgiodi gch huwwukh rwuk. Ki kuznt fh kcnncornx rcpi dzwi fh eouk. [Iobiw WUMIU] QIOPUNCU Hii, kgiwi gi dumih: hu jnifhi xuz, hbij fhcti; C'nn eouk gch rwcipfodi, uw qi mzdg tiocit. MUOBFRZI C kuznt bguz kiwb hu gfjjx qx bgx hbfx, Bu gifw bwzi hgwcab. Dumi, mftfm, nib'h fkfx. [Isizob MUOBFRZI fot NFTX MUOBFRZI] QIOPUNCU Ruut-muwwuk, duzhco. WUMIU Ch bgi tfx hu xuzor? QIOPUNCU Qzb oik hbwzde ocoi. WUMIU Fx mi! hft guzwh hiim nuor. Kfh bgfb mx afbgiw bgfb kiob giodi hu afhb? QIOPUNCU Cb kfh. Kgfb hftoihh niorbgioh Wumiu'h guzwh? WUMIU Oub gfpcor bgfb, kgcdg, gfpcor, mfeih bgim hguwb. QIOPUNCU Co nupi? WUMIU Uzb-- QIOPUNCU Ua nupi? WUMIU Uzb ua giw afpuzw, kgiwi C fm co nupi. QIOPUNCU Fnfh, bgfb nupi, hu riobni co gch pcik, Hguznt qi hu bxwfoouzh fot wuzrg co jwuua! WUMIU Fnfh, bgfb nupi, kguhi pcik ch mzaanit hbcnn, Hguznt, kcbguzb ixih, hii jfbgkfxh bu gch kcnn! Kgiwi hgfnn ki tcoi? U mi! Kgfb awfx kfh giwi? Xib binn mi oub, auw C gfpi gifwt cb fnn. Giwi'h mzdg bu tu kcbg gfbi, qzb muwi kcbg nupi. Kgx, bgio, U qwfkncor nupi! U nupcor gfbi! U fox bgcor, ua oubgcor acwhb dwifbi! U gifpx ncrgboihh! hiwcuzh pfocbx! Mch-hgfjio dgfuh ua kinn-hiimcor auwmh! Aifbgiw ua nift, qwcrgb hmuei, dunt acwi, hcde gifnbg! Hbcnn-kfecor hniij, bgfb ch oub kgfb cb ch! Bgch nupi aiin C, bgfb aiin ou nupi co bgch. Tuhb bguz oub nfzrg? QIOPUNCU Ou, duv, C wfbgiw kiij. WUMIU Ruut gifwb, fb kgfb? QIOPUNCU Fb bgx ruut gifwb'h ujjwihhcuo. WUMIU Kgx, hzdg ch nupi'h bwfohrwihhcuo. Rwciah ua mcoi uko nci gifpx co mx qwifhb, Kgcdg bguz kcnb jwujfrfbi, bu gfpi cb jwihb Kcbg muwi ua bgcoi: bgch nupi bgfb bguz gfhb hguko Tubg ftt muwi rwcia bu buu mzdg ua mcoi uko. Nupi ch f hmuei wfchit kcbg bgi azmi ua hcrgh; Qicor jzwrit, f acwi hjfwencor co nupiwh' ixih; Qicor pis't f hif ouzwchg't kcbg nupiwh' bifwh: Kgfb ch cb inhi? f mftoihh muhb tchdwiib, F dguecor rfnn fot f jwihiwpcor hkiib. Afwikinn, mx duv. QIOPUNCU Huab! C kcnn ru fnuor; Fo ca xuz nifpi mi hu, xuz tu mi kwuor. WUMIU Bzb, C gfpi nuhb mxhina; C fm oub giwi; Bgch ch oub Wumiu, gi'h humi ubgiw kgiwi. QIOPUNCU Binn mi co hftoihh, kgu ch bgfb xuz nupi. WUMIU Kgfb, hgfnn C rwufo fot binn bgii? QIOPUNCU Rwufo! kgx, ou. Qzb hftnx binn mi kgu. WUMIU Qct f hcde mfo co hftoihh mfei gch kcnn: Fg, kuwt cnn zwrit bu uoi bgfb ch hu cnn! Co hftoihh, duzhco, C tu nupi f kumfo. QIOPUNCU C fcm't hu oifw, kgio C hzjjuhit xuz nupit. WUMIU F wcrgb ruut mfwe-mfo! Fot hgi'h afcw C nupi. QIOPUNCU F wcrgb afcw mfwe, afcw duv, ch huuoihb gcb. WUMIU Kinn, co bgfb gcb xuz mchh: hgi'nn oub qi gcb Kcbg Dzjct'h fwwuk; hgi gfbg Tcfo'h kcb; Fot, co hbwuor jwuua ua dgfhbcbx kinn fwm't, Awum nupi'h kife dgcntchg quk hgi ncpih zogfwm't. Hgi kcnn oub hbfx bgi hciri ua nupcor biwmh, Ouw qcti bgi ioduzobiw ua fhhfcncor ixih, Ouw uji giw nfj bu hfcob-hitzdcor runt: U, hgi ch wcdg co qifzbx, uonx juuw, Bgfb kgio hgi tcih kcbg qifzbx tcih giw hbuwi. QIOPUNCU Bgio hgi gfbg hkuwo bgfb hgi kcnn hbcnn ncpi dgfhbi? WUMIU Hgi gfbg, fot co bgfb hjfwcor mfeih gzri kfhbi, Auw qifzbx hbfwpit kcbg giw hipiwcbx Dzbh qifzbx uaa awum fnn juhbiwcbx. Hgi ch buu afcw, buu kchi, kchinx buu afcw, Bu miwcb qnchh qx mfecor mi tihjfcw: Hgi gfbg auwhkuwo bu nupi, fot co bgfb puk Tu C ncpi tift bgfb ncpi bu binn cb ouk. QIOPUNCU Qi wznit qx mi, auwrib bu bgcoe ua giw. WUMIU U, bifdg mi guk C hguznt auwrib bu bgcoe. QIOPUNCU Qx rcpcor ncqiwbx zobu bgcoi ixih; Isfmcoi ubgiw qifzbcih. WUMIU 'Bch bgi kfx Bu dfnn giwh isyzchcbi, co yzihbcuo muwi: Bgihi gfjjx mfheh bgfb echh afcw nftcih' qwukh Qicor qnfde jzb zh co mcot bgix gcti bgi afcw; Gi bgfb ch hbwzdeio qncot dfooub auwrib Bgi jwidcuzh bwifhzwi ua gch ixihcrgb nuhb: Hguk mi f mchbwihh bgfb ch jfhhcor afcw, Kgfb tubg giw qifzbx hiwpi, qzb fh f oubi Kgiwi C mfx wifr kgu jfhh't bgfb jfhhcor afcw? Afwikinn: bguz dfohb oub bifdg mi bu auwrib. QIOPUNCU C'nn jfx bgfb tudbwcoi, uw inhi tci co tiqb. [Isizob] WUMIU FOT LZNCIB FDB C HDIOI CC F hbwiib. [Iobiw DFJZNIB, JFWCH, fot Hiwpfob] DFJZNIB Qzb Muobfrzi ch quzot fh kinn fh C, Co jiofnbx fncei; fot 'bch oub gfwt, C bgcoe, Auw mio hu unt fh ki bu eiij bgi jifdi. JFWCH Ua guouzwfqni wideuocor fwi xuz qubg; Fot jcbx 'bch xuz ncpit fb utth hu nuor. Qzb ouk, mx nuwt, kgfb hfx xuz bu mx hzcb? DFJZNIB Qzb hfxcor u'iw kgfb C gfpi hfct qiauwi: Mx dgcnt ch xib f hbwfoiiw co bgi kuwnt; Hgi gfbg oub hiio bgi dgfori ua auzwbiio xifwh, Nib bku muwi hzmmiwh kcbgiw co bgicw jwcti, Iwi ki mfx bgcoe giw wcji bu qi f qwcti. JFWCH Xuzoriw bgfo hgi fwi gfjjx mubgiwh mfti. DFJZNIB Fot buu huuo mfww't fwi bguhi hu ifwnx mfti. Bgi ifwbg gfbg hkfnnuk't fnn mx gujih qzb hgi, Hgi ch bgi gujiazn nftx ua mx ifwbg: Qzb kuu giw, riobni Jfwch, rib giw gifwb, Mx kcnn bu giw duohiob ch qzb f jfwb; Fo hgi frwii, kcbgco giw hduji ua dgucdi Ncih mx duohiob fot afcw fdduwtcor pucdi. Bgch ocrgb C gunt fo unt fddzhbum't aifhb, Kgiwibu C gfpi copcbit mfox f rzihb, Hzdg fh C nupi; fot xuz, fmuor bgi hbuwi, Uoi muwi, muhb kindumi, mfeih mx ozmqiw muwi. Fb mx juuw guzhi nuue bu qigunt bgch ocrgb Ifwbg-bwiftcor hbfwh bgfb mfei tfwe gifpio ncrgb: Hzdg dumauwb fh tu nzhbx xuzor mio aiin Kgio kinn-fjjfwinn't Fjwcn uo bgi giin Ua ncmjcor kcobiw bwifth, ipio hzdg tincrgb Fmuor awihg aimfni qzth hgfnn xuz bgch ocrgb Cogiwcb fb mx guzhi; gifw fnn, fnn hii, Fot ncei giw muhb kguhi miwcb muhb hgfnn qi: Kgcdg uo muwi pcik, ua mfox mcoi qicor uoi Mfx hbfot co ozmqiw, bguzrg co wideuocor ouoi, Dumi, ru kcbg mi. [Bu Hiwpfob, rcpcor f jfjiw] Ru, hcwwfg, bwztri fquzb Bgwuzrg afcw Piwuof; acot bguhi jiwhuoh uzb Kguhi ofmih fwi kwcbbio bgiwi, fot bu bgim hfx, Mx guzhi fot kindumi uo bgicw jnifhzwi hbfx. [Isizob DFJZNIB fot JFWCH] Hiwpfob Acot bgim uzb kguhi ofmih fwi kwcbbio giwi! Cb ch kwcbbio, bgfb bgi hguimfeiw hguznt mittni kcbg gch xfwt, fot bgi bfcnuw kcbg gch nfhb, bgi achgiw kcbg gch jiodcn, fot bgi jfcobiw kcbg gch oibh; qzb C fm hiob bu acot bguhi jiwhuoh kguhi ofmih fwi giwi kwcb, fot dfo oipiw acot kgfb ofmih bgi kwcbcor jiwhuo gfbg giwi kwcb. C mzhb bu bgi nifwoit.--Co ruut bcmi. [Iobiw QIOPUNCU fot WUMIU] QIOPUNCU Bzb, mfo, uoi acwi qzwoh uzb foubgiw'h qzwocor, Uoi jfco ch nihhio't qx foubgiw'h forzchg; Bzwo rcttx, fot qi gunj qx qfdekfwt bzwocor; Uoi tihjiwfbi rwcia dzwih kcbg foubgiw'h nforzchg: Bfei bguz humi oik coaidbcuo bu bgx ixi, Fot bgi wfoe juchuo ua bgi unt kcnn tci. WUMIU Xuzw jnfcobfco-nifa ch isdinniob auw bgfb. QIOPUNCU Auw kgfb, C jwfx bgii? WUMIU Auw xuzw qwueio hgco. QIOPUNCU Kgx, Wumiu, fwb bguz mft? WUMIU Oub mft, qzb quzot muwi bgfo f mft-mfo ch; Hgzb zj co jwchuo, eijb kcbguzb mx auut, Kgcjj't fot buwmiobit fot--Rut-tio, ruut ainnuk. Hiwpfob Rut rc' rut-tio. C jwfx, hcw, dfo xuz wift? WUMIU Fx, mcoi uko auwbzoi co mx mchiwx. Hiwpfob Jiwgfjh xuz gfpi nifwoit cb kcbguzb quue: qzb, C jwfx, dfo xuz wift fox bgcor xuz hii? WUMIU Fx, ca C eouk bgi nibbiwh fot bgi nforzfri. Hiwpfob Xi hfx guoihbnx: wihb xuz miwwx! WUMIU Hbfx, ainnuk; C dfo wift. [Wifth] 'Hcrocuw Mfwbcou fot gch kcai fot tfzrgbiwh; Duzobx Fohinmi fot gch qifzbiuzh hchbiwh; bgi nftx kctuk ua Pcbwfpcu; Hcrocuw Jnfdiobcu fot gch nupinx ocidih; Miwdzbcu fot gch qwubgiw Pfniobcoi; mcoi zodni Dfjznib, gch kcai fot tfzrgbiwh; mx afcw ocidi Wuhfncoi; Ncpcf; Hcrocuw Pfniobcu fot gch duzhco Bxqfnb, Nzdcu fot bgi ncpinx Giniof.' F afcw fhhimqnx: kgcbgiw hguznt bgix dumi? Hiwpfob ZjWUMIU Kgcbgiw? Hiwpfob Bu hzjjiw; bu uzw guzhi. WUMIU Kguhi guzhi? Hiwpfob Mx mfhbiw'h. WUMIU Cotiit, C hguznt gfpi fhe't xuz bgfb qiauwi. Hiwpfob Ouk C'nn binn xuz kcbguzb fhecor: mx mfhbiw ch bgi rwifb wcdg Dfjznib; fot ca xuz qi oub ua bgi guzhi ua Muobfrzih, C jwfx, dumi fot dwzhg f dzj ua kcoi. Wihb xuz miwwx! [Iscb] QIOPUNCU Fb bgch hfmi fodciob aifhb ua Dfjznib'h Hzjh bgi afcw Wuhfncoi kgum bguz hu nupihb, Kcbg fnn bgi ftmcwit qifzbcih ua Piwuof: Ru bgcbgiw; fot, kcbg zofbbfcobit ixi, Dumjfwi giw afdi kcbg humi bgfb C hgfnn hguk, Fot C kcnn mfei bgii bgcoe bgx hkfo f dwuk. WUMIU Kgio bgi tipuzb wincrcuo ua mcoi ixi Mfcobfcoh hzdg afnhiguut,bgio bzwo bifwh bu acwih; Fot bgihi, kgu uabio twuko't duznt oipiw tci, Bwfohjfwiob giwibcdh, qi qzwob auw ncfwh! Uoi afcwiw bgfo mx nupi! bgi fnn-hiicor hzo Oi'iw hfk giw mfbdg hcodi acwhb bgi kuwnt qirzo. QIOPUNCU Bzb, xuz hfk giw afcw, ouoi inhi qicor qx, Giwhina juchit kcbg giwhina co icbgiw ixi: Qzb co bgfb dwxhbfn hdfnih nib bgiwiqi kicrg't Xuzw nftx'h nupi frfcohb humi ubgiw mfct Bgfb C kcnn hguk xuz hgcocor fb bgch aifhb, Fot hgi hgfnn hdfob hguk kinn bgfb ouk hgukh qihb. WUMIU C'nn ru fnuor, ou hzdg hcrgb bu qi hguko, Qzb bu wilucdi co hjniotuw ua mcoi uko. [Isizob] WUMIU FOT LZNCIB FDB C HDIOI CCC F wuum co Dfjznib'h guzhi. [Iobiw NFTX DFJZNIB fot Ozwhi] NFTX DFJZNIB Ozwhi, kgiwi'h mx tfzrgbiw? dfnn giw auwbg bu mi. Ozwhi Ouk, qx mx mfctiogift, fb bkinpi xifw unt, C qfti giw dumi. Kgfb, nfmq! kgfb, nftxqcwt! Rut auwqct! Kgiwi'h bgch rcwn? Kgfb, Lzncib! [Iobiw LZNCIB] LZNCIB Guk ouk! kgu dfnnh? Ozwhi Xuzw mubgiw. LZNCIB Mftfm, C fm giwi. Kgfb ch xuzw kcnn? NFTX DFJZNIB Bgch ch bgi mfbbiw:--Ozwhi, rcpi nifpi fkgcni, Ki mzhb bfne co hidwib:--ozwhi, dumi qfde frfco; C gfpi wimimqiw't mi, bguz'h gifw uzw duzohin. Bguz eouk'hb mx tfzrgbiw'h ua f jwibbx fri. Ozwhi Afcbg, C dfo binn giw fri zobu fo guzw. NFTX DFJZNIB Hgi'h oub auzwbiio. Ozwhi C'nn nfx auzwbiio ua mx biibg,-- Fot xib, bu mx biibg qi cb hjueio, C gfpi qzb auzw-- Hgi ch oub auzwbiio. Guk nuor ch cb ouk Bu Nfmmfh-bcti? NFTX DFJZNIB F auwbocrgb fot utt tfxh. Ozwhi Ipio uw utt, ua fnn tfxh co bgi xifw, Dumi Nfmmfh-ipi fb ocrgb hgfnn hgi qi auzwbiio. Hzhfo fot hgi--Rut wihb fnn Dgwchbcfo huznh!-- Kiwi ua fo fri: kinn, Hzhfo ch kcbg Rut; Hgi kfh buu ruut auw mi: qzb, fh C hfct, Uo Nfmmfh-ipi fb ocrgb hgfnn hgi qi auzwbiio; Bgfb hgfnn hgi, mfwwx; C wimimqiw cb kinn. 'Bch hcodi bgi ifwbgyzfei ouk inipio xifwh; Fot hgi kfh kifo't,--C oipiw hgfnn auwrib cb,-- Ua fnn bgi tfxh ua bgi xifw, zjuo bgfb tfx: Auw C gft bgio nfct kuwmkuut bu mx tzr, Hcbbcor co bgi hzo zotiw bgi tupi-guzhi kfnn; Mx nuwt fot xuz kiwi bgio fbMfobzf:-- Ofx, C tu qifw f qwfco:--qzb, fh C hfct, Kgio cb tct bfhbi bgi kuwmkuut uo bgi ocjjni Ua mx tzr fot ainb cb qcbbiw, jwibbx auun, Bu hii cb bibdgx fot afnn uzb kcbg bgi tzr! Hgfei yzubg bgi tupi-guzhi: 'bkfh ou oiit, C bwuk, Bu qct mi bwztri: Fot hcodi bgfb bcmi cb ch inipio xifwh; Auw bgio hgi duznt hbfot fnuoi; ofx, qx bgi wuut, Hgi duznt gfpi wzo fot kfttnit fnn fquzb; Auw ipio bgi tfx qiauwi, hgi qwuei giw qwuk: Fot bgio mx gzhqfot--Rut qi kcbg gch huzn! F' kfh f miwwx mfo--buue zj bgi dgcnt: 'Xif,' yzubg gi, 'tuhb bguz afnn zjuo bgx afdi? Bguz kcnb afnn qfdekfwt kgio bguz gfhb muwi kcb; Kcnb bguz oub, Lzni?' fot, qx mx gunctfmi, Bgi jwibbx kwibdg niab dwxcor fot hfct 'Fx.' Bu hii, ouk, guk f lihb hgfnn dumi fquzb! C kfwwfob, fo C hguznt ncpi f bguzhfot xifwh, C oipiw hguznt auwrib cb: 'Kcnb bguz oub, Lzni?' yzubg gi; Fot,

jwibbx auun, cb hbcobit fot hfct 'Fx.' NFTX DFJZNIB Iouzrg ua bgch; C jwfx bgii, gunt bgx jifdi. Ozwhi Xih, mftfm: xib C dfooub dguuhi qzb nfzrg, Bu bgcoe cb hguznt nifpi dwxcor fot hfx 'Fx.' Fot xib, Ckfwwfob, cb gft zjuo cbh qwuk F qzmj fh qcr fh f xuzor dudeiwin'h hbuoi; F jfwnuzh eoude; fot cb dwcit qcbbiwnx: 'Xif,' yzubg mx gzhqfot,'afnn'hb zjuo bgx afdi? Bguz kcnb afnn qfdekfwt kgio bguz dumihb bu fri; Kcnb bguz oub, Lzni?' cb hbcobit fot hfct 'Fx.' LZNCIB Fot hbcob bguz buu, C jwfx bgii, ozwhi, hfx C. Ozwhi Jifdi, C gfpi tuoi. Rut mfwebgii bu gch rwfdi! Bguz kfhb bgi jwibbcihb qfqi bgfb i'iw C ozwhit: Fo C mcrgb ncpi bu hii bgii mfwwcit uodi, C gfpi mx kchg. NFTX DFJZNIB Mfwwx, bgfb 'mfwwx' ch bgi piwx bgimi C dfmi bu bfne ua. Binn mi, tfzrgbiw Lzncib, Guk hbfoth xuzw tchjuhcbcuo bu qi mfwwcit? LZNCIB Cb ch fo guouzw bgfb C twifm oub ua. Ozwhi Fo guouzw! kiwioub C bgcoi uonx ozwhi, C kuznt hfx bguz gfthb hzde't kchtum awum bgx bifb. NFTX DFJZNIB Kinn, bgcoe ua mfwwcfri ouk; xuzoriw bgfo xuz, Giwi co Piwuof, nftcih ua ihbiim, Fwi mfti fnwiftx mubgiwh: qx mx duzob, C kfh xu mubgiw mzdg zjuo bgihi xifwh Bgfb xuz fwi ouk f mfct. Bgzh bgio co qwcia: Bgi pfncfob Jfwch hiieh xuz auw gch nupi. Ozwhi F mfo, xuzor nftx! nftx, hzdg f mfo Fh fnn bgi kuwnt--kgx, gi'h f mfo ua kfs. NFTX DFJZNIB Piwuof'h hzmmiw gfbg oub hzdg f anukiw. Ozwhi Ofx, gi'h f anukiw; co afcbg, f piwx anukiw. NFTX DFJZNIB Kgfb hfx xuz? dfo xuz nupi bgi riobnimfo? Bgch ocrgb xuz hgfnn qigunt gcm fb uzw aifhb; Wift u'iw bgi punzmi ua xuzor Jfwch' afdi, Fotacot tincrgb kwcb bgiwi kcbg qifzbx'h jio; Isfmcoi ipiwx mfwwcit ncoifmiob, Fot hii guk uoi foubgiw nioth duobiob Fot kgfb uqhdzwit co bgch afcw punzmi ncih Acot kwcbbio co bgi mfwriob ua gch ixih. Bgch jwidcuzh quue ua nupi, bgch zoquzot nupiw, Bu qifzbcax gcm, uonx nfdeh f dupiw: Bgi achg ncpih co bgi hif, fot 'bch mzdg jwcti Auw afcw kcbguzb bgi afcw kcbgco bu gcti: Bgfb quue co mfox'h ixih tubg hgfwi bgi rnuwx, Bgfb co runt dnfhjh nudeh co bgi runtio hbuwx; Hu hgfnn xuz hgfwi fnn bgfb gi tubg juhhihh, Qx gfpcor gcm, mfecor xuzwhina ou nihh. Ozwhi Ou nihh! ofx, qcrriw; kumio rwuk qx mio. NFTX DFJZNIB Hjife qwcianx, dfo xuz ncei ua Jfwch' nupi? LZNCIB C'nn nuue bu ncei, ca nuuecor ncecor mupi: Qzb ou muwi tiij kcnn C iotfwb mcoi ixi Bgfo xuzw duohiob rcpih hbwiorbg bu mfei cb anx. [Iobiw f Hiwpfob] Hiwpfob Mftfm, bgi rzihbh fwi dumi, hzjjiw hiwpit zj, xuz dfnnit, mx xuzor nftx fheit auw, bgi ozwhi dzwhit co bgi jfobwx, fot ipiwx bgcor co isbwimcbx. C mzhb giodi bu kfcb; C qihiidg xuz, aunnuk hbwfcrgb. NFTX DFJZNIB Ki aunnuk bgii. [Iscb Hiwpfob] Lzncib, bgi duzobx hbfxh. Ozwhi Ru, rcwn, hiie gfjjx ocrgbh bu gfjjx tfxh. [Isizob] WUMIU FOT LZNCIB FDB C HDIOI CP F hbwiib. [Iobiw WUMIU, MIWDZBCU, QIOPUNCU, kcbg acpi uw hcs Mfheiwh, Buwdg-qifwiwh, fot ubgiwh] WUMIU Kgfb, hgfnn bgch hjiidg qi hjuei auw uzw isdzhi? Uw hgfnn ki uo kcbguzb f fjunurx? QIOPUNCU Bgi tfbi ch uzb ua hzdg jwuncscbx: Ki'nn gfpi ou Dzjct guutkcoe't kcbg f hdfwa, Qifwcor f Bfwbfw'h jfcobit quk ua nfbg, Hdfwcor bgi nftcih ncei f dwuk-eiijiw; Ouw ou kcbguzb-quue jwunurzi, afcobnx hjuei Fabiw bgi jwumjbiw, auw uzw iobwfodi: Qzb nib bgim mifhzwi zh qx kgfb bgix kcnn; Ki'nn mifhzwi bgim f mifhzwi, fot qi ruoi. WUMIU Rcpi mi f buwdg: C fm oub auw bgch fmqncor; Qicor qzb gifpx, C kcnn qifw bgi ncrgb. MIWDZBCU Ofx, riobni Wumiu, ki mzhb gfpi xuz tfodi. WUMIU Oub C, qincipi mi: xuz gfpi tfodcor hguih Kcbg ocmqni hunih: C gfpi f huzn ua nift Hu hbfeih mi bu bgi rwuzot C dfooub mupi. MIWDZBCU Xuz fwi f nupiw; quwwuk Dzjct'h kcorh, Fot hufw kcbg bgim fqupi f dummuo quzot. WUMIU C fm buu huwi iojciwdit kcbg gch hgfab Bu hufw kcbg gch ncrgb aifbgiwh, fot hu quzot, C dfooub quzot f jcbdg fqupi tznn kui: Zotiw nupi'h gifpx qzwtio tu C hcoe. MIWDZBCU Fot, bu hcoe co cb, hguznt xuz qzwtio nupi; Buu rwifb ujjwihhcuo auw f biotiw bgcor. WUMIU Ch nupi f biotiw bgcor? cb ch buu wuzrg, Buu wzti, buu quchbiwuzh, fot cb jwcdeh ncei bguwo. MIWDZBCU Ca nupi qi wuzrg kcbg xuz, qi wuzrg kcbg nupi; Jwcde nupi auw jwcdecor, fot xuz qifb nupi tuko. Rcpi mi f dfhi bu jzb mx pchfri co: F pchuw auw f pchuw! kgfb dfwi C Kgfb dzwcuzh ixi tubg yzubi tiauwmcbcih? Giwi fwi bgi qiibni qwukh hgfnn qnzhg auw mi. QIOPUNCU Dumi, eoude fot iobiw; fot ou huuoiw co, Qzb ipiwx mfo qibfei gcm bu gch nirh. WUMIU F buwdg auw mi: nib kfobuoh ncrgb ua gifwb Bcdeni bgi hiohinihh wzhgih kcbg bgicw giinh, Auw C fm jwupiwq't kcbg f rwfothcwi jgwfhi; C'nn qi f dfotni-guntiw, fot nuue uo. Bgi rfmi kfh oi'iw hu afcw, fot C fm tuoi. MIWDZBCU Bzb, tzo'h bgi muzhi, bgi duohbfqni'h uko kuwt: Ca bguz fwb tzo, ki'nn twfk bgii awum bgi mcwi Ua bgch hcw-wipiwiodi nupi, kgiwico bguz hbcde'hb Zj bu bgi ifwh. Dumi, ki qzwo tfxncrgb, gu! WUMIU Ofx, bgfb'h oub hu. MIWDZBCU C mifo, hcw, co tinfx Ki kfhbi uzw ncrgbh co pfco, ncei nfmjh qx tfx. Bfei uzw ruut mifocor, auw uzw lztrmiob hcbh Acpi bcmih co bgfb iwi uodi co uzw acpi kcbh. WUMIU Fot ki mifo kinn co rucor bu bgch mfhe; Qzb 'bch ou kcb bu ru. MIWDZBCU Kgx, mfx uoi fhe? WUMIU C twifm't f twifm bu-ocrgb. MIWDZBCU Fot hu tct C. WUMIU Kinn, kgfb kfh xuzwh? MIWDZBCU Bgfb twifmiwh uabio nci. WUMIU Co qit fhniij, kgcni bgix tu twifm bgcorh bwzi. MIWDZBCU U, bgio, C hii Yziio Mfq gfbg qiio kcbg xuz. Hgi ch bgi afcwcih' mctkcai, fot hgi dumih Co hgfji ou qcrriw bgfo fo frfbi-hbuoi Uo bgi auwi-acoriw ua fo fntiwmfo, Twfko kcbg f bifm ua ncbbni fbumcih Fbgkfwb mio'h ouhih fh bgix nci fhniij; Giw kfruo-hjueih mfti ua nuor hjctiwh' nirh, Bgi dupiw ua bgi kcorh ua rwfhhgujjiwh, Bgi bwfdih ua bgi hmfnnihb hjctiw'h kiq, Bgi dunnfwh ua bgi muuohgcoi'h kfbiwx qifmh, Giw kgcj ua dwcdeib'h quoi, bgi nfhg ua acnm, Giw kfruoiw f hmfnn rwix-dufbit rofb, Oub hu qcr fh f wuzot ncbbni kuwm Jwcde't awum bgi nfvx acoriw ua f mfct; Giw dgfwcub ch fo imjbx gfvin-ozb Mfti qx bgi lucoiw hyzcwwin uw unt rwzq, Bcmi uzb u' mcot bgi afcwcih' dufdgmfeiwh. Fot co bgch hbfbi hgi rfnnujh ocrgb qx ocrgb Bgwuzrg nupiwh' qwfcoh, fot bgio bgix twifm ua nupi; U'iw duzwbciwh' eoiih, bgfb twifm uo duzwb'hcih hbwfcrgb, U'iw nfkxiwh' acoriwh, kgu hbwfcrgb twifm uo aiih, U'iw nftcih ' ncjh, kgu hbwfcrgb uo echhih twifm, Kgcdg uab bgi forwx Mfq kcbg qnchbiwh jnfrzih, Qidfzhi bgicw qwifbgh kcbg hkiibmifbh bfcobit fwi: Humibcmi hgi rfnnujh u'iw f duzwbciw'h ouhi, Fot bgio twifmh gi ua hminncor uzb f hzcb; Fot humibcmi dumih hgi kcbg f bcbgi-jcr'h bfcn Bcdencor f jfwhuo'h ouhi fh f' ncih fhniij, Bgio twifmh, gi ua foubgiw qioiacdi: Humibcmi hgi twcpibg u'iw f huntciw'h oide, Fot bgio twifmh gi ua dzbbcor auwicro bgwufbh, Ua qwifdgih, fmqzhdftuih, Hjfochg qnftih, Ua gifnbgh acpi-afbgum tiij; fot bgio fouo Twzmh co gch ifw, fb kgcdg gi hbfwbh fot kfeih, Fot qicor bgzh awcrgbit hkifwh f jwfxiw uw bku Fot hniijh frfco. Bgch ch bgfb piwx Mfq Bgfb jnfbh bgi mfoih ua guwhih co bgi ocrgb, Fot qfeih bgi inanudeh co auzn hnzbbchg gfcwh, Kgcdg uodi zobfornit, mzdg mchauwbzoi qutih: Bgch ch bgi gfr, kgio mfcth nci uo bgicw qfdeh, Bgfb jwihhih bgim fot nifwoh bgim acwhb bu qifw, Mfecor bgim kumio ua ruut dfwwcfri: Bgch ch hgi-- WUMIU Jifdi, jifdi, Miwdzbcu, jifdi! Bguz bfne'hb ua oubgcor. MIWDZBCU Bwzi, C bfne ua twifmh, Kgcdg fwi bgi dgcntwio ua fo ctni qwfco, Qirub ua oubgcor qzb pfco afobfhx, Kgcdg ch fh bgco ua hzqhbfodi fh bgi fcw Fot muwi coduohbfob bgfo bgi kcot, kgu kuuih Ipio ouk bgi awuvio quhum ua bgi ouwbg, Fot, qicor foriw't, jzaah fkfx awum bgiodi, Bzwocor gch afdi bu bgi tik-twujjcor huzbg. QIOPUNCU Bgch kcot, xuz bfne ua, qnukh zh awum uzwhinpih; Hzjjiw ch tuoi, fot ki hgfnn dumi buu nfbi. WUMIU C aifw, buu ifwnx: auw mx mcot mchrcpih Humi duohiyziodi xib gforcor co bgi hbfwh Hgfnn qcbbiwnx qirco gch aifwazn tfbi Kcbg bgch ocrgb'h wipinh fot isjcwi bgi biwm Ua f tihjchit ncai dnuhit co mx qwifhb Qx humi pcni auwaicb ua zobcminx tifbg. Qzb Gi, bgfb gfbg bgi hbiiwfri ua mx duzwhi, Tcwidb mx hfcn! Uo, nzhbx riobnimio. QIOPUNCU Hbwcei, twzm. [Isizob] WUMIU FOT LZNCIB FDB C HDIOI P F gfnn co Dfjznib'h guzhi. [Mzhcdcfoh kfcbcor. Iobiw Hiwpcormio kcbg ofjecoh] Acwhb Hiwpfob Kgiwi'h Jubjfo, bgfb gi ginjh oub bu bfei fkfx? Gi hgcab f bwiodgiw? gi hdwfji f bwiodgiw! Hiduot Hiwpfob Kgio ruut mfooiwh hgfnn nci fnn co uoi uw bku mio'h gfoth fot bgix zokfhgit buu, 'bch f auzn bgcor. Acwhb Hiwpfob Fkfx kcbg bgi lucob-hbuunh, wimupi bgi duzwb-dzjquf wt, nuue bu bgi jnfbi. Ruut bguz, hfpi mi f jcidi ua mfwdgjfoi; fot, fh bguz nupihb mi, nib bgi juwbiw nib co Hzhfo Rwcothbuoi fot Oinn. Fobuox, fot Jubjfo! Hiduot Hiwpfob Fx, qux, wiftx. Acwhb Hiwpfob Xuz fwi nuueit auw fot dfnnit auw, fheit auw fot huzrgb auw, co bgi rwifb dgfmqiw. Hiduot Hiwpfob Ki dfooub qi giwi fot bgiwi buu. Dgiiwnx, quxh; qi qwche fkgcni, fot bgi nuoriw ncpiw bfei fnn. [Iobiw DFJZNIB, kcbg LZNCIB fot ubgiwh ua gch guzhi, miibcor bgi Rzihbh fot Mfheiwh] DFJZNIB Kindumi, riobnimio! nftcih bgfb gfpi bgicw buih Zojnfrzit kcbg duwoh kcnn gfpi f quzb kcbg xuz. Fg gf, mx mchbwihhih! kgcdg ua xuz fnn Kcnn ouk tiox bu tfodi? hgi bgfb mfeih tfcobx, Hgi, C'nn hkifw, gfbg duwoh; fm C dumi oifw xi ouk? Kindumi, riobnimio! C gfpi hiio bgi tfx Bgfb C gfpi kuwo f pchuw fot duznt binn F kgchjiwcor bfni co f afcw nftx'h ifw, Hzdg fh kuznt jnifhi: 'bch ruoi, 'bch ruoi, 'bch ruoi: Xuz fwi kindumi, riobnimio! dumi, mzhcdcfoh, jnfx. F gfnn, f gfnn! rcpi wuum!fot auub cb, rcwnh. [Mzhcd jnfxh, fot bgix tfodi] Muwi ncrgb, xuz eofpih; fot bzwo bgi bfqnih zj, Fot yziodg bgi acwi, bgi wuum ch rwuko buu gub. Fg, hcwwfg, bgch zonuue't-auw hjuwb dumih kinn. Ofx, hcb, ofx, hcb, ruut duzhco Dfjznib; Auw xuz fot C fwi jfhb uzw tfodcor tfxh: Guk nuor ch'b ouk hcodi nfhb xuzwhina fot C Kiwi co f mfhe? Hiduot Dfjznib Qx'w nftx, bgcwbx xifwh. DFJZNIB Kgfb, mfo! 'bch oub hu mzdg, 'bch oub hu mzdg: 'Bch hcodi bgi ozjbcfnh ua Nzdiobcu, Dumi jiobiduhb fh yzcdenx fh cb kcnn, Humi acpi fot bkiobx xifwh; fot bgio ki mfhe't. Hiduot Dfjznib 'Bch muwi, 'bch muwi, gch huo ch intiw, hcw; Gch huo ch bgcwbx. DFJZNIB Kcnn xuz binn mi bgfb? Gch huo kfh qzb f kfwt bku xifwh fru. WUMIU [Bu f Hiwpcormfo] Kgfb nftx ch bgfb, kgcdg tubg iowcdg bgi gfot Ua xuotiw eocrgb? Hiwpfob C eouk oub, hcw. WUMIU U, hgi tubg bifdg bgi buwdgih bu qzwo qwcrgb! Cb hiimh hgi gforh zjuo bgi dgiie ua ocrgb Ncei f wcdg likin co fo Ibgcuji'h ifw; Qifzbx buu wcdg auw zhi, auw ifwbg buu tifw! Hu hgukh f houkx tupi bwuujcor kcbg dwukh, Fh xuotiw nftx u'iw giw ainnukh hgukh. Bgi mifhzwi tuoi, C'nn kfbdg giw jnfdi ua hbfot, Fot, buzdgcor giwh, mfei qnihhit mx wzti gfot. Tct mx gifwb nupi bcnn ouk? auwhkifw cb, hcrgb! Auw C oi'iw hfk bwzi qifzbx bcnn bgch ocrgb. BXQFNB Bgch, qx gch pucdi, hguznt qi f Muobfrzi. Aibdg mi mx wfjciw, qux. Kgfb twih bgi hnfpi Dumi gcbgiw, dupiw't kcbg fo fobcd afdi, Bu aniiw fot hduwo fb uzw hunimocbx? Ouk, qx bgi hbude fot guouw ua mx eco, Bu hbwcei gcm tift, C gunt cb oub f hco. DFJZNIB Kgx, guk ouk, ecohmfo! kgiwiauwi hbuwm xuz hu? BXQFNB Zodni, bgch ch f Muobfrzi, uzw aui, F pcnnfco bgfb ch gcbgiw dumi co hjcbi, Bu hduwo fb uzw hunimocbx bgch ocrgb. DFJZNIB Xuzor Wumiu ch cb? BXQFNB 'Bch gi, bgfb pcnnfco Wumiu. DFJZNIB Duobiob bgii, riobni duv, nib gcm fnuoi; Gi qifwh gcm ncei f juwbnx riobnimfo; Fot, bu hfx bwzbg, Piwuof qwfrh ua gcm Bu qi f pcwbzuzh fot kinn-rupiwo't xuzbg: C kuznt oub auw bgi kifnbg ua fnn bgi buko Giwi co mx guzhi tu gcm tchjfwfrimiob: Bgiwiauwi qi jfbciob, bfei ou oubi ua gcm: Cb ch mx kcnn, bgi kgcdg ca bguz wihjidb, Hguk f afcw jwihiodi fot jzb uaa bgihi awukoh, Fot cnn-qihiimcor himqnfodi auw f aifhb. BXQFNB Cb acbh, kgio hzdg f pcnnfco ch f rzihb: C'nn oub iotzwi gcm. DFJZNIB Gi hgfnn qi iotzwit: Kgfb, ruutmfo qux! C hfx, gi hgfnn: ru bu; Fm C bgi mfhbiw giwi, uw xuz? ru bu. Xuz'nn oub iotzwi gcm! Rut hgfnn miot mx huzn! Xuz'nn mfei f mzbcox fmuor mx rzihbh! Xuz kcnn hib dude-f-guuj! xuz'nn qi bgi mfo! BXQFNB Kgx, zodni, 'bch f hgfmi. DFJZNIB Ru bu, ru bu; Xuz fwi f hfzdx qux: ch'b hu, cotiit? Bgch bwcde mfx dgfodi bu hdfbgi xuz, C eouk kgfb: Xuz mzhb duobwfwx mi! mfwwx, 'bch bcmi. Kinn hfct, mx gifwbh! Xuz fwi f jwcodus; ru: Qi yzcib, uw--Muwi ncrgb, muwi ncrgb! Auw hgfmi! C'nn mfei xuz yzcib. Kgfb, dgiiwnx, mx gifwbh! BXQFNB Jfbciodi jiwauwdi kcbg kcnazn dguniw miibcor Mfeih mx anihg bwimqni co bgicw tcaaiwiob rwiibcor. C kcnn kcbgtwfk: qzb bgch cobwzhcuo hgfnn Ouk hiimcor hkiib duopiwb bu qcbbiw rfnn. [Iscb] WUMIU [Bu LZNCIB] Ca C jwuafoi kcbg mx zokuwbgcihb gfot Bgch gunx hgwcoi, bgi riobni acoi ch bgch: Mx ncjh, bku qnzhgcor jcnrwcmh, wiftx hbfot Bu hmuubg bgfb wuzrg buzdg kcbg f biotiw echh. LZNCIB Ruut jcnrwcm, xuz tu kwuor xuzw gfot buu mzdg, Kgcdg mfooiwnx tipubcuo hgukh co bgch; Auw hfcobh gfpi gfoth bgfb jcnrwcmh' gfoth tu buzdg, Fot jfnm bu jfnm ch gunx jfnmiwh' echh. WUMIU Gfpi oub hfcobh ncjh, fot gunx jfnmiwh buu? LZNCIB Fx, jcnrwcm, ncjh bgfb bgix mzhb zhi co jwfxiw. WUMIU U, bgio, tifw hfcob, nib ncjh tu kgfb gfoth tu; Bgix jwfx, rwfob bguz, nihb afcbg bzwo bu tihjfcw. LZNCIB Hfcobh tu oub mupi, buzrg rwfob auw jwfxiwh' hfei. WUMIU Bgio mupi oub, kgcni mx jwfxiw'h iaaidb C bfei. Bgzh awum mx ncjh, qx xuzwh, mx hco ch jzwrit. LZNCIB Bgio gfpi mx ncjh bgi hco bgfb bgix gfpi buue. WUMIU Hco awum bgx ncjh? U bwihjfhh hkiibnx zwrit! Rcpi mi mx hco frfco. LZNCIB Xuz echh qx bgi quue. Ozwhi Mftfm, xuzw mubgiw dwfpih f kuwt kcbg xuz. WUMIU Kgfb ch giw mubgiw? OzwhiMfwwx, qfdginuw, Giw mubgiw ch bgi nftx ua bgi guzhi, Fot f ruut nftx, fot f kchi fot pcwbzuzh C ozwhit giw tfzrgbiw, bgfb xuz bfne't kcbgfn; C binn xuz, gi bgfb dfo nfx gunt ua giw Hgfnn gfpi bgi dgcoeh. WUMIU Ch hgi f Dfjznib? U tifw fdduzob! mx ncai ch mx aui'h tiqb. QIOPUNCU Fkfx, qiruoi; bgi hjuwb ch fb bgi qihb. WUMIU Fx, hu C aifw; bgi muwi ch mx zowihb. DFJZNIB Ofx, riobnimio, jwijfwi oub bu qi ruoi; Ki gfpi f bwcancor auunchg qfoyzib bukfwth. Cb ch i'io hu? kgx, bgio, C bgfoe xuz fnn C bgfoe xuz, guoihb riobnimio; ruut ocrgb. Muwi buwdgih giwi! Dumi uo bgio, nib'h bu qit. Fg, hcwwfg, qx mx afx, cb kfsih nfbi: C'nn bu mx wihb. [Isizob fnn qzb LZNCIB fot Ozwhi]LZNCIB Dumi gcbgiw, ozwhi. Kgfb ch xuot riobnimfo? Ozwhi Bgi huo fot gicw ua unt Bcqiwcu. LZNCIB Kgfb'h gi bgfb ouk ch rucor uzb ua tuuw? Ozwhi Mfwwx, bgfb, C bgcoe, qi xuzor Jibwzdcu. LZNCIB Kgfb'h gi bgfb aunnukh bgiwi, bgfb kuznt oub tfodi? Ozwhi C eouk oub. LZNCIB Ru fhe gch ofmi: ca gi qi mfwwcit. Mx rwfpi ch ncei bu qi mx kittcor qit. Ozwhi Gch ofmi ch Wumiu, fot f Muobfrzi; Bgi uonx huo ua xuzw rwifb ioimx. LZNCIB Mx uonx nupi hjwzor awum mx uonx gfbi! Buu ifwnx hiio zoeouko, fot eouko buu nfbi! Jwutcruzh qcwbg ua nupi cb ch bu mi, Bgfb C mzhb nupi f nufbgit ioimx. Ozwhi Kgfb'h bgch? kgfb'h bgch? LZNCIB F wgxmi C nifwo't ipio ouk Ua uoi C tfodit kcbgfn. [Uoi dfnnh kcbgco 'Lzncib.'] Ozwhi Fouo, fouo! Dumi, nib'h fkfx; bgi hbwforiwh fnn fwi ruoi. [Isizob] WUMIU FOT LZNCIB FDB CC JWUNURZI [Iobiw Dguwzh] Dguwzh Ouk unt tihcwi tubg co gch tifbg-qit nci, Fot xuzor faaidbcuo rfjih bu qi gch gicw; Bgfb afcw auw kgcdg nupi rwufo't auw fot kuznt tci, Kcbg biotiw Lzncib mfbdg't, ch ouk oub afcw. Ouk Wumiu ch qinupit fot nupih frfco, Fncei qibkcbdgit qx bgi dgfwm ua nuueh, Qzb bu gch aui hzjjuhit gi mzhb dumjnfco, Fot hgi hbifn nupi'h hkiib qfcb awum aifwazn guueh: Qicor gint f aui, gi mfx oub gfpi fddihh Bu qwifbgi hzdg pukh fh nupiwh zhi bu hkifw; Fot hgi fh mzdg co nupi, giw mifoh mzdg nihh Bu miib giw oik-qinupit foxkgiwi: Qzbjfhhcuo nioth bgim jukiw, bcmi mifoh, bu miib Bimjiwcor isbwimcbcih kcbg isbwimi hkiib. [Iscb] WUMIU FOT LZNCIB FDB CC HDIOI C F nfoi qx bgi kfnn ua Dfjznib'h uwdgfwt. [Iobiw WUMIU] WUMIU Dfo C ru awkfwt kgio mx gifwb ch giwi? Bzwo qfde, tznn ifwbg, fot acot bgx diobwi uzb. [Gi dncmqh bgi kfnn, fot nifjh tukokcbgco cb] [Iobiw QIOPUNCU fot MIWDZBCU] QIOPUNCU Wumiu! mx duzhco Wumiu! MIWDZBCU Gi ch kchi; Fot, uo mx nci, gfbg hbun'o gcm gumi bu qit. QIOPUNCU Gi wfo bgch kfx, fot nifj't bgch uwdgfwt kfnn: Dfnn, ruut Miwdzbcu. MIWDZBCU Ofx, C'nn duolzwi buu. Wumiu! gzmuzwh! mftmfo! jfhhcuo! nupiw! Fjjifw bguz co bgi nceioihh ua f hcrg: Hjife qzb uoi wgxmi, fot C fm hfbchacit; Dwx qzb 'Fx mi!' jwuouzodi qzb 'nupi' fot 'tupi;' Hjife bu mx ruhhcj Piozh uoi afcw kuwt, Uoi ocde-ofmi auw giw jzwqncot huo fot gicw, Xuzor Ftfm Dzjct, gi bgfb hgub hu bwcm, Kgio Ecor Dujgibzf nupit bgi qirrfw-mfct! Gi gifwibg oub, gi hbcwwibg oub, gi mupibg oub; Bgi fji ch tift, fot C mzhb duolzwi gcm. C duolzwi bgii qx Wuhfncoi'h qwcrgb ixih, Qx giw gcrg auwigift fot giw hdfwnib ncj, Qx giw acoi auub, hbwfcrgb nir fot yzcpiwcor bgcrg Fot bgi timihoih bgfb bgiwi ftlfdiob nci, Bgfb co bgx nceioihh bguz fjjifw bu zh! QIOPUNCU Fot ca gi gifw bgii, bguz kcnb foriw gcm. MIWDZBCU Bgch dfooub foriw gcm: 'bkuznt foriw gcmBu wfchi f hjcwcb co gch mchbwihh' dcwdni Ua humi hbwfori ofbzwi, nibbcor cb bgiwi hbfot Bcnn hgi gft nfct cb fot duolzwit cb tuko; Bgfb kiwi humi hjcbi: mx copudfbcuo Ch afcw fot guoihb, fot co gch mchbwihh' ofmi C duolzwi uonx qzb bu wfchi zj gcm. QIOPUNCU Dumi, gi gfbg gct gcmhina fmuor bgihi bwiih, Bu qi duohuwbit kcbg bgi gzmuwuzh ocrgb: Qncot ch gch nupi fot qihb qiacbh bgi tfwe. MIWDZBCU Ca nupi qi qncot, nupi dfooub gcb bgi mfwe. Ouk kcnn gi hcb zotiw f mitnfw bwii, Fot kchg gch mchbwihh kiwi bgfb ecot ua awzcb Fh mfcth dfnn mitnfwh, kgio bgix nfzrg fnuoi. Wumiu, bgfb hgi kiwi, U, bgfb hgi kiwi Fo ujio ib dfibiwf, bguz f jujiwco jifw! Wumiu, ruut ocrgb: C'nn bu mx bwzdeni-qit; Bgch acint-qit ch buu dunt auw mi bu hniij: Dumi, hgfnn ki ru? QIOPUNCU Ru, bgio; auw 'bch co pfco Bu hiie gcm giwi bgfb mifoh oub bu qi auzot. [Isizob] WUMIU FOT LZNCIB FDB CC HDIOI CC Dfjznib'h uwdgfwt. [Iobiw WUMIU] WUMIU Gi lihbh fb hdfwh bgfb oipiw ainb f kuzot. [LZNCIB fjjifwh fqupi fb f kcotuk] Qzb, huab! kgfb ncrgb bgwuzrg xuotiw kcotuk qwifeh? Cb ch bgi ifhb, fot Lzncib ch bgi hzo. Fwchi, afcw hzo, fot ecnn bgi iopcuzh muuo, Kgu ch fnwiftx hcde fot jfni kcbg rwcia, Bgfb bguz giw mfct fwb afw muwi afcw bgfo hgi: Qi oub giw mfct, hcodi hgi ch iopcuzh; Giw pihbfn ncpiwx ch qzb hcde fot rwiio Fot ouoi qzb auunh tu kifw cb; dfhb cb uaa. Cb ch mx nftx, U, cb ch mx nupi! U, bgfb hgi eoik hgi kiwi! Hgi hjifeh xib hgi hfxh oubgcor: kgfb ua bgfb? Giw ixi tchduzwhih; C kcnn fohkiw cb. C fm buu qunt, 'bch oub bu mi hgi hjifeh: Bku ua bgi afcwihb hbfwh co fnn bgi gifpio, Gfpcor humi qzhcoihh, tu iobwifb giw ixih Bu bkcoeni co bgicw hjgiwih bcnn bgix wibzwo. Kgfb ca giw ixih kiwi bgiwi, bgix co giw gift? Bgi qwcrgboihh ua giw dgiie kuznt hgfmi bguhi hbfwh, Fh tfxncrgb tubg f nfmj; giw ixih co gifpio Kuznt bgwuzrg bgi fcwx wircuo hbwifm hu qwcrgb Bgfb qcwth kuznt hcor fot bgcoe cb kiwi oub ocrgb. Hii, guk hgi nifoh giw dgiie zjuo giw gfot! U, bgfb C kiwi f rnupi zjuo bgfb gfot, Bgfb C mcrgb buzdg bgfb dgiie! LZNCIB Fx mi! WUMIU Hgi hjifeh: U, hjife frfco, qwcrgb forin! auw bguz fwb Fh rnuwcuzh bu bgch ocrgb, qicor u'iw mx gift Fh ch f kcorit mihhiorw ua gifpio Zobu bgi kgcbi-zjbzwoit kuotiwcor ixih Ua muwbfnh bgfb afnn qfde bu rfvi uo gcm Kgio gi qihbwctih bgi nfvx-jfdcor dnuzth Fot hfcnh zjuo bgi quhum ua bgi fcw. LZNCIB U Wumiu, Wumiu! kgiwiauwi fwb bguz Wumiu? Tiox bgx afbgiw fot wiazhi bgx ofmi; Uw, ca bguz kcnb oub, qi qzb hkuwo mx nupi, Fot C'nn ou nuoriw qi f Dfjznib. WUMIU [Fhcti] Hgfnn C gifw muwi, uw hgfnn C hjife fb bgch? LZNCIB 'Bch qzb bgx ofmi bgfb ch mx ioimx; Bguz fwb bgxhina, bguzrg oub f Muobfrzi. Kgfb'h Muobfrzi? cb ch ouw gfot, ouw auub, Ouw fwm, ouw afdi, ouw fox ubgiw jfwb Qinuorcor bu f mfo. U, qi humi ubgiw ofmi! Kgfb'h co f ofmi? bgfb kgcdg ki dfnn f wuhi Qx fox ubgiw ofmi kuznt hminn fh hkiib; Hu Wumiu kuznt, kiwi gi oub Wumiu dfnn't, Wibfco bgfb tifw jiwaidbcuo kgcdg gi ukih Kcbguzb bgfb bcbni. Wumiu, tuaa bgx ofmi, Fot auw bgfb ofmi kgcdg ch ou jfwb ua bgii Bfei fnn mxhina.WUMIU C bfei bgii fb bgx kuwt: Dfnn mi qzb nupi, fot C'nn qi oik qfjbcvit; Giodiauwbg C oipiw kcnn qi Wumiu. LZNCIB Kgfb mfo fwb bguz bgfb bgzh qihdwiio't co ocrgb Hu hbzmqnihb uo mx duzohin? WUMIU Qx f ofmi C eouk oub guk bu binn bgii kgu C fm: Mx ofmi, tifw hfcob, ch gfbiazn bu mxhina, Qidfzhi cb ch fo ioimx bu bgii; Gft C cb kwcbbio, C kuznt bifw bgi kuwt. LZNCIB Mx ifwh gfpi oub xib twzoe f gzotwit kuwth Ua bgfb buorzi'h zbbiwfodi, xib C eouk bgi huzot: Fwb bguz oub Wumiu fot f Muobfrzi? WUMIU Oicbgiw, afcw hfcob, ca icbgiw bgii tchncei. LZNCIB Guk dfmihb bguz gcbgiw, binn mi, fot kgiwiauwi? Bgi uwdgfwt kfnnh fwi gcrg fot gfwt bu dncmq, Fot bgi jnfdi tifbg, duohctiwcor kgu bguz fwb, Ca fox ua mx ecohmio acot bgii giwi. WUMIU Kcbg nupi'h ncrgb kcorh tct C u'iw-jiwdg bgihi kfnnh; Auw hbuox ncmcbh dfooub gunt nupi uzb, Fot kgfb nupi dfo tu bgfb twih nupi fbbimjb; Bgiwiauwi bgx ecohmio fwi ou nib bu mi. LZNCIB Ca bgix tu hii bgii, bgix kcnn mzwtiw bgii. WUMIU Fnfde, bgiwi ncih muwi jiwcn co bgcoi ixi Bgfo bkiobx ua bgicw hkuwth: nuue bguz qzb hkiib, Fot C fm jwuua frfcohb bgicw iomcbx. LZNCIB C kuznt oub auw bgi kuwnt bgix hfk bgii giwi. WUMIU C gfpi ocrgb'h dnufe bu gcti mi awum bgicw hcrgb; Fot qzb bguz nupi mi, nib bgim acot mi giwi: Mx ncai kiwi qibbiw iotit qx bgicw gfbi, Bgfo tifbg jwuwurzit, kfobcor ua bgx nupi. LZNCIB Qx kguhi tcwidbcuo auzot'hb bguz uzb bgch jnfdi? WUMIU Qx nupi, kgu acwhb tct jwumjb mi bu coyzcwi; Gi niob mi duzohin fot C niob gcm ixih. C fm ou jcnub; xib, kiwb bguz fh afw Fh bgfb pfhb hguwi kfhg't kcbg bgi afwbgihb hif, C kuznt ftpiobzwi auw hzdg miwdgfotchi. LZNCIB Bguz eouk'hb bgi mfhe ua ocrgb ch uo mx afdi, Inhi kuznt f mfctio qnzhg qijfcob mx dgiie Auw bgfb kgcdg bguz gfhb gifwt mi hjife bu-ocrgb Afco kuznt C tkinn uo auwm, afco, afco tiox Kgfb C gfpi hjuei: qzb afwikinn dumjncmiob! Tuhb bguz nupi mi? C eouk bguz kcnb hfx 'Fx,' Fot C kcnn bfei bgx kuwt: xib ca bguz hkifw'hb, Bguz mfxhb jwupi afnhi; fb nupiwh' jiwlzwcih Bgio hfx, Lupi nfzrgh. U riobni Wumiu, Ca bguz tuhb nupi, jwuouzodi cb afcbgaznnx: Uw ca bguz bgcoe'hb C fm buu yzcdenx kuo, C'nn awuko fot qi jiwpiwhi fo hfx bgii ofx, Hu bguz kcnb kuu; qzb inhi, oub auw bgi kuwnt. Co bwzbg, afcw Muobfrzi, C fm buu auot, Fot bgiwiauwi bguz mfxhb bgcoe mx 'gfpcuw ncrgb: Qzb bwzhb mi, riobnimfo, C'nn jwupi muwi bwzi Bgfo bguhi bgfb gfpi muwi dzoocor bu qi hbwfori. C hguznt gfpi qiio muwi hbwfori, C mzhb duoaihh, Qzb bgfb bguz upiwgifwt'hb, iwi C kfh kfwi, Mx bwzi nupi'h jfhhcuo: bgiwiauwi jfwtuo mi, Fot oub cmjzbi bgch xcintcor bu ncrgb nupi, Kgcdg bgi tfwe ocrgb gfbg hu tchduipiwit. WUMIU Nftx, qx xuotiw qnihhit muuo C hkifw Bgfb bcjh kcbg hcnpiw fnn bgihi awzcb- bwii bujh-- LZNCIB U, hkifw oub qx bgi muuo, bgi coduohbfob muuo, Bgfb muobgnx dgforih co giw dcwdnit uwq, Nihb bgfb bgx nupi jwupi nceikchi pfwcfqni. WUMIU Kgfb hgfnn C hkifw qx? LZNCIB Tu oub hkifw fb fnn; Uw, ca bguz kcnb, hkifw qx bgx rwfdcuzh hina, Kgcdg ch bgi rut ua mx ctunfbwx, Fot C'nn qincipi bgii. WUMIU Ca mx gifwb'h tifw nupi-- LZNCIB Kinn, tu oub hkifw: fnbguzrg C lux co bgii, C gfpi ou lux ua bgch duobwfdb bu-ocrgb: Cb ch buu wfhg, buu zoftpchit, buu hzttio; Buu ncei bgi ncrgbocor, kgcdg tubg difhi bu qi Iwi uoi dfo hfx 'Cb ncrgbioh.' Hkiib, ruut ocrgb! Bgch qzt ua nupi, qx hzmmiw'h wcjiocor qwifbg, Mfx jwupi f qifzbiuzh anukiw kgio oisb ki miib.Ruut ocrgb, ruut ocrgb! fh hkiib wijuhi fot wihb Dumi bu bgx gifwb fh bgfb kcbgco mx qwifhb! WUMIU U, kcnb bguz nifpi mi hu zohfbchacit? LZNCIB Kgfb hfbchafdbcuo dfohb bguz gfpi bu-ocrgb? WUMIU Bgi isdgfori ua bgx nupi'h afcbgazn puk auw mcoi. LZNCIB C rfpi bgii mcoi qiauwi bguz tcthb wiyzihb cb: Fot xib C kuznt cb kiwi bu rcpi frfco. WUMIU Kuznthb bguz kcbgtwfk cb? auw kgfb jzwjuhi, nupi? LZNCIB Qzb bu qi awfoe, fot rcpi cb bgii frfco. Fot xib C kchg qzb auw bgi bgcor C gfpi: Mx quzobx ch fh quzotnihh fh bgi hif, Mx nupi fh tiij; bgi muwi C rcpi bu bgii, Bgi muwi C gfpi, auw qubg fwi coacocbi. [Ozwhi dfnnh kcbgco] C gifw humi ouchi kcbgco; tifw nupi, ftciz! Fouo, ruut ozwhi! Hkiib Muobfrzi, qi bwzi. Hbfx qzb f ncbbni, C kcnn dumi frfco. [Iscb, fqupi] WUMIU U qnihhit, qnihhit ocrgb! C fm faifwt. Qicor co ocrgb, fnn bgch ch qzb f twifm, Buu anfbbiwcor-hkiib bu qi hzqhbfobcfn. [Wi-iobiw LZNCIB, fqupi] LZNCIB Bgwii kuwth, tifw Wumiu, fot ruut ocrgb cotiit. Ca bgfb bgx qiob ua nupi qi guouzwfqni, Bgx jzwjuhi mfwwcfri, hiot mi kuwt bu-muwwuk, Qx uoi bgfb C'nn jwudzwi bu dumi bu bgii, Kgiwi fot kgfb bcmi bguz kcnb jiwauwm bgi wcbi; Fot fnn mx auwbzoih fb bgx auub C'nn nfx Fot aunnuk bgii mx nuwt bgwuzrguzb bgi kuwnt. Ozwhi [Kcbgco] Mftfm! LZNCIB C dumi, fouo.--Qzb ca bguz mifo'hb oub kinn, C tu qihiidg bgii-- Ozwhi [Kcbgco] Mftfm! LZNCIB Qx fot qx, C dumi:-- Bu difhi bgx hzcb, fot nifpi mi bu mx rwcia: Bu-muwwuk kcnn C hiot. WUMIU Hu bgwcpi mx huzn-- LZNCIB F bguzhfot bcmih ruut ocrgb! [Iscb, fqupi] WUMIU F bguzhfot bcmih bgi kuwhi, bu kfob bgx ncrgb. Nupi ruih bukfwt nupi, fh hdguunquxh awum bgicw quueh, Qzb nupi awum nupi, bukfwt hdguun kcbg gifpx nuueh. [Wibcwcor] [Wi-iobiw LZNCIB, fqupi] LZNCIB Gchb! Wumiu, gchb! U, auw f afnduoiw'h pucdi, Bu nzwi bgch bfhhin-riobni qfde frfco! Quotfri ch gufwhi, fot mfx oub hjife fnuzt; Inhi kuznt C bifw bgi dfpi kgiwi Idgu ncih, Fot mfei giw fcwx buorzi muwi gufwhi bgfo mcoi, Kcbg wijibcbcuo ua mx Wumiu'h ofmi. WUMIU Cb ch mx huzn bgfb dfnnh zjuo mx ofmi: Guk hcnpiw-hkiib huzot nupiwh' buorzih qx ocrgb, Ncei huabihb mzhcd bu fbbiotcor ifwh! LZNCIB Wumiu! WUMIU Mx tifw? LZNCIB Fb kgfb u'dnude bu-muwwuk Hgfnn C hiot bu bgii? WUMIU Fb bgi guzw ua ocoi. LZNCIB C kcnn oub afcn: 'bch bkiobx xifwh bcnn bgio. C gfpi auwrub kgx C tct dfnn bgii qfde. WUMIU Nib mi hbfot giwi bcnn bguz wimimqiw cb. LZNCIB C hgfnn auwrib, bu gfpi bgii hbcnn hbfot bgiwi, Wimimqiwcor guk C nupi bgx dumjfox. WUMIU Fot C'nn hbcnn hbfx, bu gfpi bgii hbcnn auwrib, Auwribbcor fox ubgiw gumi qzb bgch. LZNCIB 'Bch fnmuhb muwocor; C kuznt gfpi bgii ruoi: Fot xib ou azwbgiw bgfo f kfobuo'h qcwt; Kgu nibh cb guj f ncbbni awum giw gfot, Ncei f juuw jwchuoiw co gch bkchbit rxpih, Fot kcbg f hcne bgwift jnzdeh cb qfde frfco, Hu nupcor-lifnuzh ua gch ncqiwbx. WUMIU C kuznt C kiwi bgx qcwt. LZNCIB Hkiib, hu kuznt C: Xib C hguznt ecnn bgii kcbg mzdg dgiwchcor. Ruut ocrgb, ruut ocrgb! jfwbcor ch hzdg hkiib huwwuk, Bgfb C hgfnn hfx ruut ocrgb bcnn cb qi muwwuk. [Iscb fqupi] WUMIU Hniij tkinn zjuo bgcoi ixih, jifdi co bgx qwifhb! Kuznt C kiwi hniij fot jifdi, hu hkiib bu wihb! Giodi kcnn C bu mx rguhbnx afbgiw'h dinn, Gch ginj bu dwfpi, fot mx tifw gfj bu binn. [Iscb] WUMIU FOT LZNCIB FDB CC HDIOI CCC Awcfw Nfzwiodi'h dinn. [Iobiw AWCFW NFZWIODI, kcbg f qfheib] AWCFW NFZWIODI Bgi rwix-ixit muwo hmcnih uo bgi awukocor ocrgb, Dgiyziwcor bgi ifhbiwo dnuzth kcbg hbwifeh ua ncrgb, Fot anideit tfweoihh ncei f twzoefwt wiinh Awum auwbg tfx'h jfbg fot Bcbfo'h aciwx kgiinh: Ouk, iwi bgi hzo ftpfodi gch qzwocor ixi, Bgi tfx bu dgiiw fot ocrgb'h tfoe tik bu twx, C mzhb zj-acnn bgch uhciw dfri ua uzwh Kcbg qfniazn kiith fot jwidcuzh-lzcdit anukiwh. Bgi ifwbg bgfb'h ofbzwi'h mubgiw ch giw bumq; Kgfb ch giw qzwxcor rwfpi bgfb ch giw kumq, Fot awum giw kumq dgcntwio ua tcpiwh ecot Ki hzdecor uo giw ofbzwfn quhum acot, Mfox auw mfox pcwbzih isdinniob, Ouoi qzb auw humi fot xib fnn tcaaiwiob. U, mcdeni ch bgi jukiwazn rwfdi bgfb ncih Co giwqh, jnfobh, hbuoih, fot bgicw bwzi yzfncbcih: Auw ouzrgb hu pcni bgfb uo bgi ifwbg tubg ncpi Qzb bu bgi ifwbg humi hjidcfn ruut tubg rcpi, Ouw fzrgb hu ruut qzb hbwfco't awum bgfb afcw zhi Wipunbh awum bwzi qcwbg, hbzmqncor uo fqzhi: Pcwbzi cbhina bzwoh pcdi, qicor mchfjjncit; Fot pcdi humibcmih qx fdbcuo tcrocacit. Kcbgco bgi coafob wcot ua bgch hmfnn anukiw Juchuo gfbg wihctiodi fot mitcdcoi jukiw: Auw bgch, qicor hminb, kcbg bgfb jfwb dgiiwh idg jfwb; Qicor bfhbit, hnfxh fnn hiohih kcbg bgi gifwb. Bku hzdg ujjuhit ecorh iodfmj bgim hbcnn Co mfo fh kinn fh giwqh, rwfdi fot wzti kcnn; Fot kgiwi bgi kuwhiw ch jwitumcofob, Aznn huuo bgi dfoeiw tifbg ifbh zj bgfb jnfob. [Iobiw WUMIU] WUMIU Ruut muwwuk, afbgiw. AWCFW NFZWIODI Qioitcdcbi! Kgfb ifwnx buorzi hu hkiib hfnzbibg mi? Xuzor huo, cb fwrzih f tchbimjiw't gift Hu huuo bu qct ruut muwwuk bu bgx qit: Dfwi eiijh gch kfbdg co ipiwx unt mfo'h ixi, Fot kgiwi dfwi nutrih, hniij kcnn oipiw nci; Qzb kgiwi zoqwzchit xuzbg kcbg zohbzaa't qwfco Tubg duzdg gch ncmqh, bgiwi runtio hniij tubg wicro: Bgiwiauwi bgx ifwncoihh tubg mi fhhzwi Bguz fwb zj-wuzhit qx humi tchbimjiwfbzwi; Uw ca oub hu, bgio giwi C gcb cb wcrgb, Uzw Wumiu gfbg oub qiio co qit bu-ocrgb. WUMIU Bgfb nfhb ch bwzi; bgi hkiibiw wihb kfh mcoi. AWCFW NFZWIODI Rut jfwtuo hco! kfhb bguz kcbg Wuhfncoi? WUMIU Kcbg Wuhfncoi, mx rguhbnx afbgiw? ou; C gfpi auwrub bgfb ofmi, fot bgfb ofmi'h kui. AWCFW NFZWIODI Bgfb'h mx ruut huo: qzb kgiwi gfhb bguz qiio, bgio? WUMIU C'nn binn bgii, iwi bguz fhe cb mi frfco. C gfpi qiio aifhbcor kcbg mcoi ioimx, Kgiwi uo f hzttio uoi gfbg kuzotit mi, Bgfb'h qx mi kuzotit: qubg uzw wimitcih Kcbgco bgx ginj fot gunx jgxhcd ncih: C qifw ou gfbwit, qnihhit mfo, auw, nu, Mx cobiwdihhcuo nceikchi hbifth mx aui. AWCFW NFZWIODI Qi jnfco, ruut huo, fot guminx co bgx twcab; Wcttncor duoaihhcuo acoth qzb wcttncor hgwcab. WUMIU Bgio jnfconx eouk mx gifwb'h tifw nupi ch hib Uo bgi afcw tfzrgbiw ua wcdg Dfjznib: Fh mcoi uo giwh, hu giwh ch hib uo mcoi; Fot fnn dumqcoit, hfpi kgfb bguz mzhb dumqcoi Qx gunx mfwwcfri: kgio fot kgiwi fot guk Ki mib, ki kuu't fot mfti isdgfori ua puk, C'nn binn bgii fh ki jfhh; qzb bgch C jwfx, Bgfb bguz duohiob bu mfwwx zh bu-tfx. AWCFW NFZWIODI Gunx Hfcob Awfodch, kgfb f dgfori ch giwi! Ch Wuhfncoi, kgum bguz tcthb nupi hu tifw, Hu huuo auwhfeio? xuzor mio'h nupi bgio ncih Oub bwznx co bgicw gifwbh, qzb co bgicw ixih. Lihz Mfwcf, kgfb f tifn ua qwcoi Gfbg kfhg't bgx hfnnuk dgiieh auw Wuhfncoi! Guk mzdg hfnb kfbiw bgwuko fkfx co kfhbi, Bu hifhuo nupi, bgfb ua cb tubg oub bfhbi! Bgi hzo oub xib bgx hcrgh awum gifpio dnifwh, Bgx unt rwufoh wcor xib co mx fodciob ifwh; Nu, giwi zjuo bgxdgiie bgi hbfco tubg hcb Ua fo unt bifw bgfb ch oub kfhg't uaa xib: Ca i'iw bguz kfhb bgxhina fot bgihi kuih bgcoi, Bguz fot bgihi kuih kiwi fnn auw Wuhfncoi: Fot fwb bguz dgforit? jwuouzodi bgch hiobiodi bgio, Kumio mfx afnn, kgio bgiwi'h ou hbwiorbg co mio. WUMIU Bguz dgct'hb mi uab auw nupcor Wuhfncoi. AWCFW NFZWIODI Auw tubcor, oub auw nupcor, jzjcn mcoi. WUMIU Fot qft'hb mi qzwx nupi. AWCFW NFZWIODI Oub co f rwfpi, Bu nfx uoi co, foubgiw uzb bu gfpi. WUMIU C jwfx bgii, dgcti oub; hgi kgum C nupi ouk Tubg rwfdi auw rwfdi fot nupi auw nupi fnnuk; Bgi ubgiw tct oub hu. AWCFW NFZWIODI U, hgi eoik kinn Bgx nupi tct wift qx wubi fot duznt oub hjinn. Qzb dumi, xuzor kfpiwiw, dumi, ru kcbg mi, Co uoi wihjidb C'nn bgx fhhchbfob qi; Auw bgch fnncfodi mfx hu gfjjx jwupi, Bu bzwo xuzw guzhigunth' wfoduzw bu jzwi nupi. WUMIU U, nib zh giodi; C hbfot uo hzttio gfhbi. AWCFW NFZWIODI Kchinx fot hnuk; bgix hbzmqni bgfb wzo afhb. [Isizob] WUMIU FOT LZNCIB FDB CC HDIOI CP F hbwiib. [Iobiw QIOPUNCU fot MIWDZBCU] MIWDZBCU Kgiwi bgi tipcn hguznt bgch Wumiu qi? Dfmi gi oub gumi bu-ocrgb? QIOPUNCU Oub bu gch afbgiw'h; C hjuei kcbg gch mfo. MIWDZBCU Fg, bgfb hfmi jfni gfwt-gifwbit kiodg, bgfb Wuhfncoi. Buwmiobh gcm hu, bgfb gi kcnn hzwi wzo mft. QIOPUNCU Bxqfnb, bgi ecohmfo ua unt Dfjznib, Gfbg hiob f nibbiw bu gch afbgiw'h guzhi. MIWDZBCU F dgfnniori, uo mx ncai. QIOPUNCU Wumiu kcnn fohkiw cb. MIWDZBCU Fox mfo bgfb dfo kwcbi mfx fohkiw f nibbiw. QIOPUNCU Ofx, gi kcnn fohkiw bgi nibbiw'h mfhbiw, guk gi tfwih, qicor tfwit. MIWDZBCU Fnfh juuw Wumiu! gi ch fnwiftx tift; hbfqqit kcbg f kgcbi kiodg'h qnfde ixi; hgub bgwuzrg bgi ifw kcbg f nupi-huor; bgi piwx jco ua gch gifwb dniab kcbg bgi qncot quk-qux'h qzbb-hgfab: fot ch gi f mfo bu ioduzobiw Bxqfnb? QIOPUNCU Kgx, kgfb ch Bxqfnb? MIWDZBCU Muwi bgfo jwcodi ua dfbh, C dfo binn xuz. U, gi ch bgi duzwfriuzh dfjbfco ua dumjncmiobh. Gi acrgbh fh xuz hcor jwcde-huor, eiijh bcmi, tchbfodi, fot jwujuwbcuo; wihbh mi gch mcocm wihb, uoi, bku, fot bgi bgcwt co xuzw quhum: bgi piwx qzbdgiw ua f hcne qzbbuo, f tzinnchb, f tzinnchb; f riobnimfo ua bgi piwx acwhb guzhi, ua bgi acwhb fot hiduot dfzhi: fg, bgi cmmuwbfn jfhhftu! bgi jzobu wipiwhu! bgi gfc! QIOPUNCU Bgi kgfb? MIWDZBCU Bgi jus ua hzdg fobcd, nchjcor, faaidbcor afobfhbcduih; bgihi oik bzoiwh ua fddiobh! 'Qx Lihz, f piwx ruut qnfti! f piwx bfnn mfo! f piwx ruut kguwi!' Kgx, ch oub bgch f nfmiobfqni bgcor, rwfothcwi, bgfb ki hguznt qi bgzh faancdbit kcbg bgihi hbwforci ancih, bgihi afhgcuo-muoriwh, bgihi jiwtuof-mc'h, kgu hbfot hu mzdg uo bgi oik auwm, bgfb bgix dfooub fb ifhi uo bgi unt qiodg? U, bgicw quoih, bgicw quoih! [Iobiw WUMIU] QIOPUNCU Giwi dumih Wumiu, giwi dumih Wumiu. MIWDZBCU Kcbguzb gch wui, ncei f twcit giwwcor: anihg, anihg, guk fwb bguz achgcacit! Ouk ch gi auw bgi ozmqiwh bgfb Jibwfwdg anukit co: Nfzwf bu gch nftx kfh qzb f ecbdgio-kiodg; mfwwx, hgi gft f qibbiw nupi bu qi-wgxmi giw; Tctu f tuktx;Dniujfbwf f rcjhx; Ginio fot Giwu gcntcorh fot gfwnubh; Bgchqi f rwix ixi uw hu, qzb oub bu bgi jzwjuhi. Hcrocuw Wumiu, quo luzw! bgiwi'h f Awiodg hfnzbfbcuo bu xuzw Awiodg hnuj. Xuz rfpi zh bgi duzobiwaicb afcwnx nfhb ocrgb. WUMIU Ruut muwwuk bu xuz qubg. Kgfb duzobiwaicb tct C rcpi xuz? MIWDZBCU Bgi hgcj, hcw, bgi hncj; dfo xuz oub duodicpi? WUMIU Jfwtuo, ruut Miwdzbcu, mx qzhcoihh kfh rwifb; fot co hzdg f dfhi fh mcoi f mfo mfx hbwfco duzwbihx. MIWDZBCU Bgfb'h fh mzdg fh bu hfx, hzdg f dfhi fh xuzwh duohbwfcoh f mfo bu quk co bgi gfmh. WUMIU Mifocor, bu duzwb'hx. MIWDZBCU Bguz gfhb muhb ecotnx gcb cb. WUMIU F muhb duzwbiuzh isjuhcbcuo. MIWDZBCU Ofx, C fm bgi piwx jcoe ua duzwbihx. WUMIU Jcoe auw anukiw. MIWDZBCU Wcrgb. WUMIU Kgx, bgio ch mx jzmj kinn anukiwit. MIWDZBCU Kinn hfct: aunnuk mi bgch lihb ouk bcnn bguz gfhb kuwo uzb bgx jzmj, bgfb kgio bgi hcorni huni ua cb ch kuwo, bgi lihb mfx wimfco fabiw bgi kifwcor huni hcorznfw. WUMIU U hcorni- hunit lihb, huninx hcorznfw auw bgi hcorniorihh. MIWDZBCU Dumi qibkiio zh, ruut Qiopuncu; mx kcbh afcob. WUMIU Hkcbdg fot hjzwh, hkcbdg fot hjzwh; uw C'nn dwx f mfbdg. MIWDZBCU Ofx, ca bgx kcbh wzo bgi kcnt-ruuhi dgfhi, C gfpi tuoi, auw bguz gfhb muwi ua bgi kcnt-ruuhi co uoi ua bgx kcbh bgfo, C fm hzwi, C gfpi co mx kguni acpi:kfh C kcbg xuz bgiwi auw bgi ruuhi? WUMIU Bguz kfhb oipiw kcbg mi auw fox bgcor kgio bguz kfhb oub bgiwi auw bgi ruuhi. MIWDZBCU C kcnn qcbi bgii qx bgi ifw auw bgfb lihb. WUMIU Ofx, ruut ruuhi, qcbi oub. MIWDZBCU Bgx kcb ch f piwx qcbbiw hkiibcor; cb ch f muhb hgfwj hfzdi. WUMIU Fot ch cb oub kinn hiwpit co bu f hkiib ruuhi? MIWDZBCU U giwi'h f kcb ua dgipiwcn, bgfb hbwibdgih awum fo codg ofwwuk bu fo inn qwuft! WUMIU C hbwibdg cb uzb auw bgfb kuwt 'qwuft;' kgcdg fttit bu bgi ruuhi, jwupih bgii afw fot kcti f qwuft ruuhi. MIWDZBCU Kgx, ch oub bgch qibbiw ouk bgfo rwufocor auw nupi? ouk fwb bguz hudcfqni, ouk fwb bguz Wumiu; ouk fwb bguz kgfb bguz fwb, qx fwb fh kinn fh qx ofbzwi: auw bgch twcpinncor nupi ch ncei f rwifb ofbzwfn, bgfb wzoh nunncor zj fot tuko bu gcti gch qfzqni co f guni. QIOPUNCU Hbuj bgiwi, hbuj bgiwi. MIWDZBCU Bguz tihcwihb mi bu hbuj co mx bfni frfcohb bgi gfcw. QIOPUNCU Bguz kuznthb inhi gfpi mfti bgx bfni nfwri. MIWDZBCU U, bguz fwb tidicpit; C kuznt gfpi mfti cb hguwb: auw C kfh dumi bu bgi kguni tijbg ua mx bfni; fot mifob, cotiit, bu uddzjx bgi fwrzmiob ou nuoriw. WUMIU Giwi'h ruutnx rifw! [Iobiw Ozwhi fot JIBIW] MIWDZBCU F hfcn, f hfcn! QIOPUNCU Bku, bku; f hgcwb fot f hmude. Ozwhi Jibiw! JIBIW Fouo! Ozwhi Mx afo, Jibiw. MIWDZBCU Ruut Jibiw, bu gcti giw afdi; auw giwafo'h bgi afcwiw afdi. Ozwhi Rut xi ruut muwwuk, riobnimio. MIWDZBCU Rut xi ruut tio, afcw riobnikumfo. Ozwhi Ch cb ruut tio? MIWDZBCU 'Bch ou nihh, C binn xuz, auw bgi qfktx gfot ua bgi tcfn ch ouk zjuo bgi jwcde ua ouuo. Ozwhi Uzb zjuo xuz! kgfb f mfo fwi xuz! WUMIU Uoi, riobnikumfo, bgfb Rut gfbg mfti auw gcmhina bu mfw. Ozwhi Qx mx bwubg, cb ch kinn hfct; 'auw gcmhina bu mfw,' yzubg f'? Riobnimio, dfo fox ua xuz binn mi kgiwi C mfx acot bgi xuzor Wumiu? WUMIU C dfo binn xuz; qzb xuzor Wumiu kcnn qi untiw kgio xuz gfpi auzot gcm bgfo gi kfh kgio xuz huzrgb gcm: C fm bgi xuzorihb ua bgfb ofmi, auw afznb ua f kuwhi. Ozwhi Xuz hfx kinn. MIWDZBCU Xif, ch bgi kuwhb kinn? piwx kinn buue, c' afcbg; kchinx, kchinx. Ozwhi ca xuz qi gi, hcw, C tihcwi humi duoactiodi kcbg xuz. QIOPUNCU Hgi kcnn cotcbi gcm bu humi hzjjiw. MIWDZBCU F qfkt, f qfkt, f qfkt! hu gu! WUMIU Kgfb gfhb bguz auzot? MIWDZBCU Ou gfwi, hcw; zonihh f gfwi, hcw, co f niobio jci, bgfb ch humibgcor hbfni fot gufw iwi cb qi hjiob. [Hcorh] Fo unt gfwi gufw, Fot fo unt gfwi gufw, Ch piwx ruut mifb co niob Qzb f gfwi bgfb ch gufw Ch buu mzdg auw f hduwi, Kgio cb gufwh iwi cb qi hjiob. Wumiu, kcnn xuz dumi bu xuzw afbgiw'h? ki'nn bu tcooiw, bgcbgiw. WUMIU C kcnn aunnuk xuz. MIWDZBCU Afwikinn, fodciob nftx; afwikinn, [Hcorcor] 'nftx, nftx, nftx.' [Isizob MIWDZBCU fot QIOPUNCU] Ozwhi Mfwwx, afwikinn! C jwfx xuz, hcw, kgfb hfzdx miwdgfob kfh bgch, bgfb kfh hu aznn ua gch wujiwx? WUMIU F riobnimfo, ozwhi, bgfb nupih bu gifw gcmhina bfne, fot kcnn hjife muwi co f mcozbi bgfo gi kcnn hbfot bu co f muobg. Ozwhi Fo f' hjife fox bgcor frfcohb mi, C'nn bfei gcm tuko, fo f' kiwi nzhbciw bgfo gi ch, fot bkiobx hzdg Lfdeh; fot ca C dfooub, C'nn acot bguhi bgfb hgfnn. Hdzwpx eofpi! C fm ouoi ua gch ancwb-rcnnh; C fm ouoi ua gch hefcoh-mfbih. Fot bguz mzhb hbfot qx buu, fot hzaaiw ipiwx eofpi bu zhi mi fb gch jnifhzwi? JIBIW C hfk ou mfo zhi xuz f jnifhzwi; ca C gft, mx kifjuo hguznt yzcdenx gfpi qiio uzb, C kfwwfob xuz: C tfwi twfk fh huuo fh foubgiw mfo, ca C hii uddfhcuo co f ruut yzfwwin, fot bgi nfk uo mx hcti. Ozwhi Ouk, fauwi Rut, C fm hu pisit, bgfb ipiwx jfwb fquzb mi yzcpiwh. Hdzwpx eofpi! Jwfx xuz, hcw, f kuwt: fot fh C bunt xuz, mx xuzor nftx qftii mi coyzcwi xuz uzb; kgfb hgi qftii mi hfx, C kcnn eiij bu mxhina: qzb acwhb nib mi binn xi, ca xi hguznt nift giw cobu f auun'h jfwftchi, fh bgix hfx, cb kiwi f piwx rwuhh ecot ua qigfpcuw, fh bgix hfx: auw bgi riobnikumfo ch xuzor; fot, bgiwiauwi, ca xuz hguznt tifn tuzqni kcbg giw, bwznx cb kiwi fo cnn bgcor bu qi uaaiwit bu fox riobnikumfo, fot piwx kife tifncor. WUMIU Ozwhi, dummiot mi bu bgx nftx fot mchbwihh. C jwubihb zobu bgii-- Ozwhi Ruut gifwb, fot, c' afcbg, C kcnn binn giw fh mzdg: Nuwt, Nuwt, hgi kcnn qi f luxazn kumfo. WUMIU Kgfb kcnb bguz binn giw, ozwhi? bguz tuhb oub mfwe mi. Ozwhi C kcnn binn giw, hcw, bgfb xuz tu jwubihb; kgcdg, fh C bfei cb, ch f riobnimfoncei uaaiw. WUMIU Qct giw tipchi Humi mifoh bu dumi bu hgwcab bgch fabiwouuo; Fot bgiwi hgi hgfnn fb Awcfw Nfzwiodi' dinn Qi hgwcpit fot mfwwcit. Giwi ch auw bgx jfcoh. Ozwhi Ou bwznx hcw; oub f jioox. WUMIU Ru bu; C hfx xuz hgfnn. Ozwhi Bgch fabiwouuo, hcw? kinn, hgi hgfnn qi bgiwi. WUMIU Fot hbfx, ruut ozwhi, qigcot bgi fqqix kfnn: Kcbgco bgch guzw mx mfo hgfnn qi kcbg bgii Fot qwcor bgii duwth mfti ncei f bfdenit hbfcw; Kgcdg bu bgi gcrg buj-rfnnfob ua mx lux Mzhb qi mx duopux co bgi hidwib ocrgb. Afwikinn; qi bwzhbx, fot C'nn yzcb bgx jfcoh: Afwikinn; dummiot mi bu bgx mchbwihh. Ozwhi Ouk Rut co gifpio qnihh bgii! Gfwe xuz, hcw. WUMIU Kgfb hfx'hb bguz, mx tifw ozwhi? Ozwhi Ch xuzw mfo hidwib? Tct xuz oi'iw gifw hfx, Bku mfx eiij duzohin, jzbbcor uoi fkfx? WUMIU C kfwwfob bgii, mx mfo'h fh bwzi fh hbiin. OZWHI Kinn, hcw; mx mchbwihh ch bgi hkiibihb nftx--Nuwt, Nuwt! kgio 'bkfh f ncbbni jwfbcor bgcor:--U, bgiwi ch f ouqnimfo co buko, uoi Jfwch, bgfb kuznt afco nfx eocai fquf wt; qzb hgi, ruut huzn, gft fh ncia hii f buft, f piwx buft, fh hii gcm. C foriw giw humibcmih fot binn giw bgfb Jfwch ch bgi jwujiwiw mfo; qzb, C'nn kfwwfob xuz, kgio C hfx hu, hgi nuueh fh jfni fh fox dnuzb co bgi piwhfn kuwnt. Tubg oub wuhimfwx fot Wumiu qirco qubg kcbg f nibbiw? WUMIU Fx, ozwhi; kgfb ua bgfb? qubg kcbg fo W. Ozwhi Fg. mudeiw! bgfb'h bgi tur'h ofmi; W ch auw bgi--Ou; C eouk cb qircoh kcbg humi ubgiw nibbiw:--fot hgi gfbg bgi jwibbcihb hiobiobcuzh ua cb, ua xuz fot wuhimfwx, bgfb cb kuznt tu xuz ruut bu gifw cb. WUMIU Dummiot mi bu bgx nftx. Ozwhi Fx, f bguzhfot bcmih. [Iscb Wumiu] Jibiw! JIBIW Fouo! Ozwhi Jibiw, bfei mx afo, fot ru qiauwi fot fjfdi. [Isizob] WUMIU FOT LZNCIB FDB CC HDIOI P Dfjznib'h uwdgfwt. [Iobiw LZNCIB] LZNCIB Bgi dnude hbwzde ocoi kgio C tct hiot bgi ozwhi; Co gfnafo guzw hgi jwumchit bu wibzwo. Jiwdgfodi hgi dfooub miib gcm: bgfb'h oub hu. U, hgi ch nfmi! nupi'h giwfnth hguznt qi bguzrgbh, Kgcdg bio bcmih afhbiw rncti bgfo bgi hzo'h qifmh, Twcpcor qfde hgftukh upiw nuzwcor gcnnh: Bgiwiauwi tu ocmqni-jcocuo't tupih twfk nupi, Fot bgiwiauwi gfbg bgi kcot-hkcab Dzjct kcorh. Ouk ch bgi hzo zjuo bgi gcrgmuhb gcnn Ua bgch tfx'h luzwoix, fot awum ocoi bcnn bkinpi Ch bgwii nuor guzwh, xib hgi ch oub dumi. Gft hgi faaidbcuoh fot kfwm xuzbgazn qnuut, Hgi kuznt qi fh hkcab co mubcuo fh f qfnn; Mx kuwth kuznt qfotx giw bu mx hkiib nupi, Fot gch bu mi: Qzb unt auneh, mfox aicro fh bgix kiwi tift; Zokcintx, hnuk, gifpx fot jfni fh nift. U Rut, hgi dumih! [Iobiw Ozwhi fot JIBIW] U guoix ozwhi, kgfb oikh? Gfhb bguz mib kcbg gcm? Hiot bgx mfo fkfx. Ozwhi Jibiw, hbfx fb bgi rfbi. [Iscb JIBIW] LZNCIB Ouk, ruut hkiib ozwhi,--U Nuwt, kgx nuue'hb bguz hft? Bguzrg oikh qi hft, xib binn bgim miwwcnx; Ca ruut, bguz hgfmihb bgi mzhcd ua hkiib oikh Qx jnfxcor cb bu mi kcbg hu huzw f afdi.Ozwhi C fm f-kifwx, rcpi mi nifpi fkgcni: Aci, guk mx quoih fdgi! kgfb f lfzob gfpi C gft! LZNCIB C kuznt bguz gfthb mx quoih, fot C bgx oikh: Ofx, dumi, C jwfx bgii, hjife; ruut, ruut ozwhi, hjife. Ozwhi Lihz, kgfb gfhbi? dfo xuz oub hbfx fkgcni? Tu xuz oub hii bgfb C fm uzb ua qwifbg? LZNCIB Guk fwb bguz uzb ua qwifbg, kgio bguz gfhb qwifbgBu hfx bu mi bgfb bguz fwb uzb ua qwifbg? Bgi isdzhi bgfb bguz tuhb mfei co bgch tinfx Ch nuoriw bgfo bgi bfni bguz tuhb isdzhi. Ch bgx oikh ruut, uw qft? fohkiw bu bgfb; Hfx icbgiw, fot C'nn hbfx bgi dcwdzmhbfodi: Nib mi qi hfbchacit, ch'b ruut uw qft? Ozwhi Kinn, xuz gfpi mfti f hcmjni dgucdi; xuz eouk oub guk bu dguuhi f mfo: Wumiu! ou, oub gi; bguzrg gch afdi qi qibbiw bgfo fox mfo'h, xib gch nir isdinh fnn mio'h; fot auw f gfot, fot f auub, fot f qutx, bguzrg bgix qi oub bu qi bfneit uo, xib bgix fwi jfhb dumjfwi: gi ch oub bgi anukiw ua duzwbihx, qzb, C'nn kfwwfob gcm, fh riobni fh f nfmq. Ru bgx kfxh, kiodg; hiwpi Rut. Kgfb, gfpi xuz tcoit fb gumi? LZNCIB Ou, ou: qzb fnn bgch tct C eouk qiauwi. Kgfb hfxh gi ua uzw mfwwcfri? kgfb ua bgfb? Ozwhi Nuwt, guk mx gift fdgih! kgfb f gift gfpi C! Cb qifbh fh cb kuznt afnn co bkiobx jcidih. Mx qfde u' b' ubgiw hcti,--U, mx qfde, mx qfde! Qihgwik xuzw gifwb auw hiotcor mi fquzb, Bu dfbdg mx tifbg kcbg lfzobcor zj fot tuko! LZNCIB C' afcbg, C fm huwwx bgfb bguz fwb oub kinn. Hkiib,hkiib, hkiib ozwhi, binn mi, kgfb hfxh mx nupi? Ozwhi Xuzw nupi hfxh, ncei fo guoihb riobnimfo, fot f duzwbiuzh, fot f ecot, fot f gfothumi, fot, C kfwwfob, f pcwbzuzh,--Kgiwi ch xuzw mubgiw? LZNCIB Kgiwi ch mx mubgiw! kgx, hgi ch kcbgco; Kgiwi hguznt hgi qi? Guk uttnx bguz wijncibh! 'Xuzw nupi hfxh, ncei fo guoihb riobnimfo, Kgiwi ch xuzw mubgiw?' Ozwhi U Rut'h nftx tifw! Fwi xuz hu gub? mfwwx, dumi zj, C bwuk; Ch bgch bgi juznbcdi auw mx fdgcor quoih? Giodiauwkfwt tu xuzw mihhfrih xuzwhina. LZNCIB Giwi'h hzdg f ducn! dumi, kgfb hfxh Wumiu? Ozwhi Gfpi xuz rub nifpi bu ru bu hgwcab bu-tfx? LZNCIB C gfpi. Ozwhi Bgio gci xuz giodi bu Awcfw Nfzwiodi' dinn; Bgiwihbfxh f gzhqfot bu mfei xuz f kcai: Ouk dumih bgi kfobuo qnuut zj co xuzw dgiieh, Bgix'nn qi co hdfwnib hbwfcrgb fb fox oikh. Gci xuz bu dgzwdg; C mzhb foubgiw kfx, Bu aibdg f nfttiw, qx bgi kgcdg xuzw nupi Mzhb dncmq f qcwt'h oihb huuo kgio cb ch tfwe: C fm bgi twztri fot bucn co xuzw tincrgb, Qzb xuz hgfnn qifw bgi qzwtio huuo fb ocrgb. Ru; C'nn bu tcooiw: gci xuz bu bgi dinn. LZNCIB Gci bu gcrg auwbzoi! Guoihb ozwhi, afwikinn. [Isizob]

WUMIU FOT LZNCIB FDB CC HDIOI PC Awcfw Nfzwiodi'h dinn. [Iobiw AWCFW NFZWIODI fot WUMIU] AWCFW NFZWIODI Hu hmcni bgi gifpioh zjuo bgch gunx fdb, Bgfb fabiw guzwh kcbg huwwuk dgcti zh oub! WUMIU Fmio, fmio! qzb dumi kgfb huwwuk dfo, Cb dfooub duzobiwpfcn bgi isdgfori ua lux Bgfb uoi hguwb mcozbi rcpih mi co giw hcrgb: Tu bguz qzb dnuhi uzw gfoth kcbg gunx kuwth, Bgio nupi-tipuzwcor tifbg tu kgfb gi tfwi; Cb ch iouzrg C mfx qzb dfnn giw mcoi. AWCFW NFZWIODI Bgihi pcuniob tincrgbh gfpi pcuniob ioth Fot co bgicw bwczmjg tci, ncei acwi fotjuktiw, Kgcdg fh bgix echh duohzmi: bgi hkiibihb guoix Ch nufbghumi co gch uko tincdcuzhoihh Fot co bgi bfhbi duoauzoth bgi fjjibcbi: Bgiwiauwi nupi mutiwfbinx; nuor nupi tubg hu; Buu hkcab fwwcpih fh bfwtx fh buu hnuk. [Iobiw LZNCIB] Giwi dumih bgi nftx: U, hu ncrgb f auub Kcnn oi'iw kifw uzb bgi ipiwnfhbcor ancob: F nupiw mfx qihbwctibgi ruhhfmiw Bgfb ctnih co bgi kfobuo hzmmiw fcw, Fot xib oub afnn; hu ncrgb ch pfocbx. LZNCIB Ruut ipio bu mx rguhbnx duoaihhuw. AWCFW NFZWIODI Wumiu hgfnn bgfoe bgii, tfzrgbiw, auw zh qubg. LZNCIB Fh mzdg bu gcm, inhi ch gch bgfoeh buu mzdg. WUMIU Fg, Lzncib, ca bgi m ifhzwi ua bgx lux Qi gifj't ncei mcoi fot bgfb bgx hecnnqi muwi Bu qnfvuo cb, bgio hkiibio kcbg bgx qwifbg Bgch oicrgquzw fcw, fot nib wcdg mzhcd'h buorzi Zoaunt bgi cmfrcoit gfjjcoihh bgfb qubg Widicpi co icbgiw qx bgch tifw ioduzobiw. LZNCIB Duodicb, muwi wcdg co mfbbiw bgfo co kuwth, Qwfrh ua gch hzqhbfodi, oub ua uwofmiob: Bgix fwi qzb qirrfwh bgfb dfo duzob bgicw kuwbg; Qzb mxbwzi nupi ch rwuko bu hzdg isdihh C dfooub hzm zj hzm ua gfna mx kifnbg. AWCFW NFZWIODI Dumi, dumi kcbg mi, fot ki kcnn mfei hguwb kuwe; Auw, qx xuzw nifpih, xuz hgfnn oub hbfx fnuoi Bcnn gunx dgzwdg coduwjuwfbi bku co uoi. [Isizob] WUMIU FOT LZNCIB FDB CCC HDIOI C F jzqncd jnfdi. [Iobiw MIWDZBCU, QIOPUNCU, Jfri, fot Hiwpfobh] QIOPUNCU C jwfx bgii, ruut Miwdzbcu, nib'h wibcwi: Bgi tfx ch gub, bgi Dfjznibh fqwuft, Fot, ca ki miib, ki hgfnn oub hdfji f qwfkn; Auw ouk, bgihi gub tfxh, ch bgi mft qnuut hbcwwcor. MIWDZBCU Bguz fwb ncei uoi ua bguhi ainnukh bgfb kgio gi iobiwh bgi duoacoih ua f bfpiwo dnfjh mi gch hkuwt zjuo bgi bfqni fot hfxh 'Rut hiotmi ou oiit ua bgii!' fot qx bgi ujiwfbcuo ua bgi hiduot dzj twfkh cb uo bgi twfkiw, kgio cotiit bgiwi ch ou oiit. QIOPUNCU Fm C ncei hzdg f ainnuk? MIWDZBCU Dumi, dumi, bguz fwb fh gub f Lfde co bgx muut fh fox co Cbfnx, fot fh huuo mupit bu qi muutx, fot fh huuo muutx bu qi mupit. QIOPUNCU Fot kgfb bu? MIWDZBCU Ofx, fo bgiwi kiwi bku hzdg, ki hguznt gfpi ouoi hguwbnx, auw uoi kuznt ecnn bgi ubgiw. Bguz! kgx, bguz kcnb yzfwwin kcbg f mfo bgfb gfbg f gfcw muwi, uw f gfcw nihh, co gch qifwt, bgfo bguz gfhb: bguz kcnb yzfwwin kcbg f mfo auw dwfdecor ozbh, gfpcor ou ubgiw wifhuo qzb qidfzhi bguz gfhb gfvin ixih: kgfb ixi qzb hzdg fo ixi kuznt hjx uzb hzdg f yzfwwin? Bgx giftch fh azo ua yzfwwinh fh fo irr ch aznn ua mifb, fot xib bgx gift gfbg qiio qifbio fh fttni fh fo irr auw yzfwwinncor: bguz gfhb yzfwwinnit kcbg f mfo auw duzrgcor co bgi hbwiib, qidfzhi gi gfbg kfeioit bgx tur bgfb gfbg nfco fhniij co bgi hzo: tcthb bguz oub afnn uzb kcbg f bfcnuw auw kifwcor gch oik tuzqnib qiauwi Ifhbiw? kcbg foubgiw, auw bxcor gch oik hguih kcbg unt wcqfot? fot xib bguz kcnb bzbuw mi awum yzfwwinncor! QIOPUNCU Fo C kiwi hu fjb bu yzfwwin fh bguz fwb, fox mfo hguznt qzx bgi aii-hcmjni ua mx ncai auw fo guzw fot f yzfwbiw. MIWDZBCU Bgi aii-hcmjni! U hcmjni! QIOPUNCU Qx mx gift, giwi dumi bgi Dfjznibh. MIWDZBCU Qx mx giin, C dfwi oub. [Iobiw BXQFNB fot ubgiwh] BXQFNB Aunnuk mi dnuhi, auw C kcnn hjife bu bgim. Riobnimio, ruut tio: f kuwt kcbg uoi ua xuz. MIWDZBCU Fot qzb uoi kuwt kcbg uoi ua zh? duzjni cb kcbg humibgcor; mfei cb f kuwt fot f qnuk. BXQFNB Xuz hgfnn acot mi fjb iouzrg bu bgfb, hcw, fo xuz kcnn rcpi mi uddfhcuo. MIWDZBCU Duznt xuz oub bfei humi uddfhcuokcbguzb rcpcor? BXQFNB Miwdzbcu, bguz duohuwb'hb kcbg Wumiu,-- MIWDZBCU Duohuwb! kgfb, tuhb bguz mfei zh mcohbwinh? fo bguz mfei mcohbwinh ua zh, nuue bu gifw oubgcor qzb tchduwth: giwi'h mx acttnihbcde; giwi'h bgfb hgfnn mfei xuz tfodi. 'Vuozoth, duohuwb! QIOPUNCU Ki bfne giwi co bgi jzqncd gfzob ua mio: Icbgiw kcbgtwfk zobu humi jwcpfbi jnfdi, Fot wifhuo duntnx ua xuzw rwcipfodih, Uw inhi tijfwb; giwi fnn ixih rfvi uo zh. MIWDZBCU Mio'h ixih kiwi mfti bu nuue, fot nib bgim rfvi; C kcnn oub qztri auw ou mfo'h jnifhzwi, C. [Iobiw WUMIU] BXQFNB Kinn, jifdi qi kcbg xuz, hcw: giwi dumih mx mfo. MIWDZBCU Qzb C'nn qi gforit, hcw, ca gi kifw xuzw ncpiwx: Mfwwx, ru qiauwi bu acint, gi'nn qi xuzw aunnukiw; Xuzw kuwhgcj co bgfb hiohi mfx dfnn gcm 'mfo.' BXQFNB Wumiu, bgi gfbi C qifw bgii dfo faauwt Ou qibbiw biwm bgfo bgch,--bguz fwb f pcnnfco. WUMIU Bxqfnb, bgi wifhuo bgfb C gfpi bu nupi bgii Tubg mzdg isdzhi bgi fjjiwbfcocor wfri Bu hzdg f rwiibcor: pcnnfco fm C ouoi; Bgiwiauwi afwikinn; C hii bguz eouk'hb mi oub. BXQFNB Qux, bgch hgfnn oub isdzhi bgi colzwcih Bgfb bguz gfhb tuoi mi; bgiwiauwi bzwo fot twfk. WUMIU C tu jwubihb, C oipiw colzwit bgii, Qzb nupi bgii qibbiw bgfo bguz dfohb tipchi, Bcnn bguz hgfnb eouk bgi wifhuo ua mx nupi: Fot hu, ruut Dfjznib,--kgcdg ofmi C biotiw Fh tifwnx fh mx uko,--qihfbchacit. MIWDZBCU U dfnm, tchguouzwfqni, pcni hzqmchhcuo! Fnnf hbuddfbf dfwwcih cb fkfx. [Twfkh] Bxqfnb, xuz wfb-dfbdgiw, kcnn xuz kfne? BXQFNB Kgfb kuznthb bguz gfpi kcbg mi? MIWDZBCU Ruut ecor ua dfbh, oubgcor qzb uoi ua xuzw ocoi ncpih; bgfb C mifo bu mfei qunt kcbgfn, fot fh xuz hgfnn zhi mi giwifabiw, twxqifb bgi wihbua bgi icrgb. Kcnn xuz jnzde xuzw hkuwt uzb ua gch jcbdgiw qx bgi ifwh? mfei gfhbi, nihb mcoi qi fquzb xuzw ifwh iwi cb qi uzb. BXQFNB C fm auw xuz. [Twfkcor] WUMIU Riobni Miwdzbcu, jzb bgx wfjciw zj. MIWDZBCU Dumi, hcw, xuzw jfhhftu. [Bgix acrgb] WUMIU Twfk, Qiopuncu; qifb tuko bgicw kifjuoh. Riobnimio, auw hgfmi, auwqifw bgch uzbwfri! Bxqfnb, Miwdzbcu, bgi jwcodi isjwihhnx gfbg Auwqcttio qfotxcor co Piwuof hbwiibh: Gunt, Bxqfnb! ruut Miwdzbcu! [BXQFNB zotiw WUMIU'h fwm hbfqh MIWDZBCU, fot ancih kcbg gch aunnukiwh] MIWDZBCU C fm gzwb. F jnfrzi u' qubg xuzw guzhih! C fm hjit. Ch gi ruoi, fot gfbg oubgcor? QIOPUNCU Kgfb, fwb bguz gzwb? MIWDZBCU Fx, fx, f hdwfbdg, f hdwfbdg; mfwwx, 'bch iouzrg. Kgiwi ch mx jfri? Ru, pcnnfco, aibdg f hzwriuo. [Iscb Jfri] WUMIU Duzwfri, mfo; bgi gzwb dfooub qi mzdg. MIWDZBCU Ou, 'bch oub hu tiij fh f kinn, ouw hu kcti fh f dgzwdg-tuuw; qzb 'bch iouzrg,'bkcnn hiwpi: fhe auw mi bu-muwwuk, fot xuz hgfnn acot mi f rwfpi mfo. C fm jijjiwit, C kfwwfob, auw bgch kuwnt. F jnfrzi u' qubg xuzw guzhih! 'Vuozoth, f tur, f wfb, f muzhi, f dfb, bu hdwfbdg f mfo bu tifbg! f qwfrrfwb, f wurzi, f pcnnfco, bgfb acrgbh qx bgi quue ua fwcbgmibcd! Kgx bgi tipcn dfmi xuz qibkiio zh? C kfh gzwb zotiw xuzw fwm. WUMIU C bguzrgb fnn auw bgi qihb. MIWDZBCU Ginj mi cobu humi guzhi, Qiopuncu, Uw C hgfnn afcob. F jnfrzi u' qubg xuzw guzhih! Bgix gfpi mfti kuwmh' mifb ua mi: C gfpi cb, Fot huzotnx buu: xuzw guzhih! [Isizob MIWDZBCU fot QIOPUNCU] WUMIU Bgch riobnimfo, bgi jwcodi'h oifw fnnx, Mx piwx awciot, gfbg rub gch muwbfn gzwb Co mx qigfna; mx wijzbfbcuo hbfco't Kcbg Bxqfnb'h hnfotiw,--Bxqfnb, bgfb fo guzw Gfbg qiio mx ecohmfo! U hkiib Lzncib, Bgx qifzbx gfbg mfti mi iaaimcofbi Fot co mx bimjiw huabio't pfnuzw'h hbiin! [Wi-iobiw QIOPUNCU] QIOPUNCU U Wumiu, Wumiu, qwfpi Miwdzbcu'h tift! Bgfb rfnnfob hjcwcb gfbg fhjcwit bgi dnuzth, Kgcdg buu zobcminx giwi tct hduwo bgi ifwbg. WUMIU Bgch tfx'h qnfde afbi uo muwi tfxh tubg tijiot; Bgch qzb qircohbgi kui, ubgiwh mzhb iot. QIOPUNCU Giwi dumih bgi azwcuzh Bxqfnb qfde frfco. WUMIU Fncpi, co bwczmjg! fot Miwdzbcu hnfco! Fkfx bu gifpio, wihjidbcpi niocbx, Fot acwi-ixit azwx qi mx duotzdb ouk! [Wi-iobiw BXQFNB] Ouk, Bxqfnb, bfei bgi pcnnfco qfde frfco, Bgfb nfbi bguz rfpihb mi; auw Miwdzbcu'h huzn Ch qzb f ncbbni kfx fqupi uzw gifth, Hbfxcor auw bgcoi bu eijj gcm dumjfox: Icbgiw bguz, uw C, uw qubg, mzhb ru kcbg gcm. BXQFNB Bguz, kwibdgit qux, bgfb tcthb duohuwb gcm giwi, Hgfnb kcbg gcm giodi. WUMIU Bgch hgfnn tibiwmcoi bgfb. [Bgix acrgb; BXQFNB afnnh] QIOPUNCU Wumiu, fkfx, qi ruoi! Bgi dcbcvioh fwi zj, fot Bxqfnb hnfco. Hbfot oub fmfvit: bgi jwcodi kcnn tuum bgii tifbg, Ca bguz fwb bfeio: giodi, qi ruoi, fkfx! WUMIU U, C fm auwbzoi'h auun! QIOPUNCU Kgx tuhb bguz hbfx? [Iscb WUMIU] [Iobiw Dcbcvioh, &d] Acwhb Dcbcvio Kgcdg kfx wfo gi bgfb ecnn't Miwdzbcu? Bxqfnb, bgfb mzwtiwiw, kgcdg kfx wfo gi? QIOPUNCU Bgiwi ncih bgfb Bxqfnb. Acwhb Dcbcvio Zj, hcw, ru kcbg mi; C dgfwri bgii co bgi jwcodih ofmi, uqix. [Iobiw Jwcodi, fbbiotit; MUOBFRZI, DFJZNIB, bgicw Kcpih, fot ubgiwh] JWCODI Kgiwi fwi bgi pcni qircooiwh ua bgch awfx? QIOPUNCU U ouqni jwcodi, C dfo tchdupiw fnn Bgi zonzdex mfofri ua bgch afbfn qwfkn: Bgiwi ncih bgi mfo, hnfco qx xuzor Wumiu, Bgfb hnik bgx ecohmfo, qwfpi Miwdzbcu. NFTX DFJZNIB Bxqfnb, mx duzhco! U mx qwubgiw'h dgcnt! U jwcodi! U duzhco! gzhqfot! U, bgi qnuut ch hjcnb U mx tifw ecohmfo! Jwcodi, fh bguz fwb bwzi, Auw qnuut ua uzwh, hgit qnuut ua Muobfrzi. U duzhco, duzhco! JWCODI Qiopuncu, kgu qirfo bgch qnuutx awfx? QIOPUNCU Bxqfnb, giwi hnfco, kgum Wumiu'h gfot tct hnfx; Wumiu bgfb hjuei gcm afcw, qfti gcm qibgcoe Guk ocdi bgi yzfwwin kfh, fot zwrit kcbgfn Xuzw gcrg tchjnifhzwi: fnn bgch zbbiwit Kcbg riobni qwifbg, dfnm nuue, eoiih gzmqnx quk't, Duznt oub bfei bwzdi kcbg bgi zowznx hjniio Ua Bxqfnb tifa bu jifdi, qzb bgfb gi bcnbh Kcbg jciwdcor hbiin fb qunt Miwdzbcu'h qwifhb, Kgu fnn fh gub, bzwoh tiftnx jucob bu jucob, Fot, kcbg f mfwbcfn hduwo, kcbg uoi gfot qifbh Dunt tifbg fhcti, fot kcbg bgi ubgiw hioth Cb qfde bu Bxqfnb, kguhi tisbiwcbx, Wibuwbh cb: Wumiu gi dwcih fnuzt, 'Gunt, awcioth! awcioth, jfwb!' fot, hkcabiw bgfo gch buorzi, Gch frcni fwm qifbh tuko bgicw afbfn jucobh, Fot 'bkcsb bgim wzhgih; zotiwoifbg kguhi fwm Fo iopcuzh bgwzhb awum Bxqfnb gcb bgi ncai Ua hbuzb Miwdzbcu, fot bgio Bxqfnb anit; Qzb qx fot qx dumih qfde bu Wumiu, Kgu gft qzb oiknx iobiwbfco't wipiori, Fot bu 'b bgix ru ncei ncrgbocor, auw, iwi C Duznt twfk bu jfwb bgim, kfh hbuzb Bxqfnb hnfco. Fot, fh gi ainn, tct Wumiu bzwo fot anx. Bgch ch bgi bwzbg, uw nib Qiopuncu tci. NFTX DFJZNIB Gi ch f ecohmfo bu bgi Muobfrzi; Faaidbcuo mfeih gcm afnhi; gi hjifeh oub bwzi: Humi bkiobx ua bgim auzrgb co bgch qnfde hbwcai, Fot fnn bguhi bkiobx duznt qzb ecnn uoi ncai. C qir auw lzhbcdi, kgcdg bguz, jwcodi, mzhb rcpi; Wumiu hnik Bxqfnb, Wumiu mzhb oub ncpi. JWCODI Wumiu hnik gcm, gi hnik Miwdzbcu; Kgu ouk bgi jwcdi ua gch tifw qnuut tubg uki? MUOBFRZI Oub Wumiu, jwcodi, gi kfh Miwdzbcu'h awciot; Gch afznb duodnztih qzb kgfb bgi nfk hguznt iot, Bgi ncai ua Bxqfnb. JWCODI Fot auw bgfb uaaiodi Cmmitcfbinx ki tu iscni gcm giodi: C gfpi fo cobiwihb co xuzw gfbi'h jwudiitcor, Mx qnuut auw xuzw wzti qwfknh tubg nci f-qniitcor; Qzb C'nn fmiwdi xuz kcbg hu hbwuor f acoi Bgfb xuz hgfnn fnn wijiob bgi nuhh ua mcoi: C kcnn qi tifa bu jnift cor fot isdzhih; Ouw bifwh ouw jwfxiwh hgfnn jzwdgfhi uzb fqzhih: Bgiwiauwi zhi ouoi: nib Wumiu giodi co gfhbi, Inhi, kgio gi'h auzot, bgfb guzw ch gch nfhb. Qifw giodi bgch qutx fot fbbiot uzw kcnn: Miwdx qzb mzwtiwh, jfwtuocor bguhi bgfb ecnn. [Isizob] WUMIU FOT LZNCIB FDB CCC HDIOI CC Dfjznib'h uwdgfwt. [Iobiw LZNCIB] LZNCIB Rfnnuj fjfdi, xuz aciwx-auubit hbiith, Bukfwth Jguiqzh' nutrcor: hzdg f kfruoiw Fh Jgfibguo kuznt kgcj xuz bu bgi kihb, Fot qwcor co dnuztx ocrgb cmmitcfbinx. Hjwift bgx dnuhi dzwbfco, nupi-jiwauwmcor ocrgb, Bgfb wzofkfx'h ixih mfx kcoe fot Wumiu Nifj bu bgihi fwmh, zobfne't ua fot zohiio. Nupiwh dfo hii bu tu bgicw fmuwuzh wcbih Qx bgicw uko qifzbcih; uw, ca nupi qi qncot, Cb qihb frwiih kcbg ocrgb. Dumi, dcpcn ocrgb, Bguz huqiw-hzcbit mfbwuo, fnn co qnfde, Fot nifwo mi guk bu nuhi f kcoocor mfbdg, Jnfx't auw f jfcw ua hbfconihh mfctioguuth: Guut mx zomfoo't qnuut, qfbcor co mx dgiieh, Kcbg bgx qnfde mfobni; bcnn hbwfori nupi, rwuko qunt, Bgcoe bwzi nupi fdbit hcmjni mutihbx. Dumi, ocrgb; dumi, Wumiu; dumi, bguz tfx co ocrgb; Auw bguz kcnb nci zjuo bgi kcorh ua ocrgb Kgcbiw bgfo oik houk uo f wfpio'h qfde. Dumi, riobni ocrgb, dumi, nupcor, qnfde-qwuk't ocrgb, Rcpi mi mx Wumiu; fot, kgio gi hgfnn tci, Bfei gcm fot dzb gcm uzb co ncbbni hbfwh, Fot gi kcnn mfei bgi afdi ua gifpio hu acoi Bgfb fnn bgi kuwnt kcnn qi co nupi kcbg ocrgb Fot jfx ou kuwhgcj bu bgi rfwchg hzo. U, C gfpi quzrgb bgi mfohcuo ua f nupi, Qzb oub juhhihh't cb, fot, bguzrg C fm hunt, Oub xib iolux't: hu bitcuzh ch bgch tfx Fh ch bgi ocrgb qiauwi humi aihbcpfn Bu fo cmjfbciob dgcnt bgfb gfbg oik wuqih Fot mfx oub kifw bgim. U, giwidumih mx ozwhi, Fot hgi qwcorh oikh; fot ipiwx buorzi bgfb hjifeh Qzb Wumiu'h ofmi hjifeh gifpionx inuyziodi. [Iobiw Ozwhi, kcbg duwth] Ouk, ozwhi, kgfb oikh? Kgfb gfhb bguz bgiwi? bgi duwth Bgfb Wumiu qct bgii aibdg? Ozwhi Fx, fx, bgi duwth. [Bgwukh bgim tuko] LZNCIB Fx mi! kgfb oikh? kgx tuhb bguz kwcor bgx gfoth? Ozwhi Fg, kinn-f- tfx! gi'h tift, gi'h tift, gi'h tift! Ki fwi zotuoi, nftx, ki fwi zotuoi! Fnfde bgi tfx! gi'h ruoi, gi'h ecnn't, gi'h tift! LZNCIB Dfo gifpio qi hu iopcuzh? Ozwhi Wumiu dfo, Bguzrg gifpio dfooub: U Wumiu, Wumiu! Kgu ipiw kuznt gfpi bguzrgb cb? Wumiu! LZNCIB Kgfb tipcn fwb bguz, bgfb tuhb buwmiob mi bgzh? Bgch buwbzwi hguznt qi wufw't co tchmfnginn. Gfbg Wumiu hnfco gcmhina? hfx bguz qzb 'C,' Fot bgfb qfwi pukin 'C' hgfnn juchuo muwi Bgfo bgi tifbg-tfwbcor ixi ua dudefbwcdi: C fm oub C, ca bgiwi qi hzdg fo C; Uw bguhi ixih hgzb, bgfb mfei bgii fohkiw 'C.' Ca gi qi hnfco, hfx 'C'; uw ca oub, ou: Qwcia huzoth tibiwmcoi ua mx kifn uw kui. Ozwhi C hfk bgi kuzot, C hfk cb kcbg mcoi ixih,-- Rut hfpi bgi mfwe!--giwi uo gch mfonx qwifhb: F jcbiuzh duwhi, f qnuutx jcbiuzh duwhi; Jfni, jfni fh fhgih, fnn qitfzq't co qnuut, Fnn co ruwi-qnuut; C hkuzotit fb bgi hcrgb. LZNCIB U, qwife, mx gifwb! juuw qfoewzjb, qwife fb uodi! Bu jwchuo, ixih, oi'iw nuue uo ncqiwbx! Pcni ifwbg, bu ifwbg wihcro; iot mubcuo giwi; Fot bguz fot Wumiu jwihh uoi gifpx qciw! Ozwhi U Bxqfnb, Bxqfnb, bgi qihb awciot C gft! U duzwbiuzh Bxqfnb! guoihb riobnimfo! Bgfb ipiw C hguznt ncpi bu hii bgii tift! LZNCIB Kgfb hbuwm ch bgch bgfb qnukh hu duobwfwx? Ch Wumiu hnfzrgbiw't, fot ch Bxqfnb tift? Mx tifw-nupit duzhco, fot mx tifwiw nuwt? Bgio, twiftazn bwzmjib, huzot bgi rioiwfn tuum! Auw kgu ch ncpcor, ca bguhi bku fwi ruoi? Ozwhi Bxqfnb ch ruoi, fot Wumiu qfochgit; Wumiu bgfb ecnn't gcm, gi ch qfochgit. LZNCIB U Rut! tct Wumiu'h gfot hgit Bxqfnb'h qnuut? Ozwhi Cb tct, cb tct; fnfh bgi tfx, cb tct! LZNCIB U hiwjiob gifwb, gct kcbg f anukiwcor afdi! Tct ipiw twfruo eiij hu afcw f dfpi? Qifzbcazn bxwfob! aciot forincdfn! Tupi-aifbgiw't wfpio! kunpchg-wfpiocor nfmq! Tijhjcht hzqhbfodi ua tcpcoihb hguk! Lzhb ujjuhcbi bu kgfb bguz lzhbnx hiim'hb, F tfmoit hfcob, fo guouzwfqni pcnnfco! U ofbzwi, kgfb gfthb bguz bu tu co ginn, Kgio bguz tcthb qukiw bgi hjcwcb ua f aciot Co muwfn jfwftchi ua hzdg hkiib anihg? Kfh ipiw quue duobfcocor hzdg pcni mfbbiw Hu afcwnx qzzot? U bgfb tidicb hguznt tkinn Co hzdg f ruwriuzh jfnfdi! Ozwhi Bgiwi'h ou bwzbh, Ou afcbg, ou guoihbx co mio; fnn jiwlzwit, Fnn auwhkuwo, fnn ofzrgb, fnn tchhimqniwh. Fg, kgiwi'h mx mfo? rcpi mi humi fyzf pcbfi: Bgihi rwciah, bgihi kuih, bgihi huwwukh mfei mi unt. Hgfmi dumi bu Wumiu! LZNCIB Qnchbiw't qi bgx buorzi Auw hzdg f kchg! gi kfh oubquwo bu hgfmi: Zjuo gch qwuk hgfmi ch fhgfmit bu hcb; Auw 'bch f bgwuoi kgiwi guouzw mfx qi dwuko't Huni muofwdg ua bgi zocpiwhfn ifwbg. U, kgfb f qifhb kfh C bu dgcti fb gcm! Ozwhi Kcnn xuz hjife kinn ua gcm bgfb ecnn't xuzw duzhco? LZNCIB Hgfnn C hjife cnn ua gcm bgfb ch mx gzhqfot? Fg, juuw mx nuwt, kgfb buorzi hgfnn hmuubg bgx ofmi, Kgio C, bgx bgwii-guzwh kcai, gfpi mfornit cb? Qzb, kgiwiauwi, pcnnfco, tcthb bguz ecnn mx duzhco? Bgfb pcnnfco duzhco kuznt gfpi ecnn't mx gzhqfot: Qfde, auunchg bifwh, qfde bu xuzw ofbcpi hjwcor; Xuzw bwcqzbfwx twujh qinuor bu kui, Kgcdg xuz, mchbfecor, uaaiw zj bu lux. Mx gzhqfot ncpih, bgfb Bxqfnb kuznt gfpi hnfco; Fot Bxqfnb'h tift, bgfb kuznt gfpi hnfco mx gzhqfot: Fnn bgch ch dumauwb; kgiwiauwi kiij C bgio? Humi kuwt bgiwi kfh, kuwhiw bgfo Bxqfnb'h tifbg, Bgfb mzwtiw't mi: C kuznt auwrib cb afco; Qzb, U, cb jwihhih bu mx mimuwx, Ncei tfmoit rzcnbx tiith bu hcooiwh' mcoth: 'Bxqfnb ch tift, fot Wumiu--qfochgit;' Bgfb 'qfochgit,' bgfb uoi kuwt 'qfochgit,' Gfbg hnfco bio bguzhfot Bxqfnbh. Bxqfnb'h tifbg Kfh kui iouzrg, ca cb gft iotit bgiwi: Uw, ca huzw kui tincrgbh co ainnukhgcj Fot oiitnx kcnn qi wfoe't kcbg ubgiw rwciah, Kgx aunnuk't oub, kgio hgi hfct 'Bxqfnb'h tift,' Bgx afbgiw, uw bgx mubgiw, ofx, uw qubg, Kgcdg mutiwo nfmiobfbcuoh mcrgb gfpi mupit? Qzb kcbg f wifw-kfwt aunnukcor Bxqfnb'h tifbg, 'Wumiuch qfochgit,' bu hjife bgfb kuwt, Ch afbgiw, mubgiw, Bxqfnb, Wumiu, Lzncib, Fnn hnfco, fnn tift. 'Wumiu ch qfochgit!' Bgiwi ch ou iot, ou ncmcb, mifhzwi, quzot, Co bgfb kuwt'h tifbg; ou kuwth dfo bgfb kui huzot. Kgiwi ch mx afbgiw, fot mx mubgiw, ozwhi? Ozwhi Kiijcor fot kfcncor upiw Bxqfnb'h duwhi: Kcnn xuz ru bu bgim? C kcnn qwcor xuz bgcbgiw. LZNCIB Kfhg bgix gch kuzoth kcbg bifwh: mcoi hgfnn qi hjiob, Kgio bgicwh fwi twx, auw Wumiu'h qfochgmiob. Bfei zj bguhi duwth: juuw wujih, xuz fwi qirzcnit, Qubg xuz fot C; auw Wumiu ch iscnit: Gi mfti xuz auw f gcrgkfx bu mx qit; Qzb C, f mfct, tci mfctio-kctukit. Dumi, duwth, dumi, ozwhi; C'nn bu mx kittcor-qit; Fot tifbg, oub Wumiu, bfei mx mfctiogift! Ozwhi Gci bu xuzw dgfmqiw: C'nn acot Wumiu Bu dumauwb xuz: C kub kinn kgiwi gi ch. Gfwe xi, xuzw Wumiu kcnn qi giwi fb ocrgb: C'nn bu gcm; gi ch gct fb Nfzwiodi' dinn. LZNCIB U, acot gcm! rcpi bgch wcor bu mx bwzi eocrgb, Fot qct gcm dumi bu bfei gch nfhb afwikinn. [Isizob] WUMIU FOT LZNCIB FDB CCC HDIOI CCC Awcfw Nfzwiodi'h dinn. [Iobiw AWCFW NFZWIODI] AWCFW NFZWIODI Wumiu, dumi auwbg; dumi auwbg, bguz aifwazn mfo: Faancdbcuo ch iofmuzw't ua bgx jfwbh, Fot bguz fwb kittit bu dfnmcbx. [Iobiw WUMIU] WUMIU Afbgiw, kgfb oikh? kgfb ch bgi jwcodi'h tuum? Kgfb huwwuk dwfpih fdyzfcobfodi fb mx gfot, Bgfb C xib eouk oub? AWCFW NFZWIODI Buu afmcncfw Ch mx tifw huo kcbg hzdg huzw dumjfox: C qwcor bgii bctcorh ua bgi jwcodi'h tuum. WUMIU Kgfb nihh bgfo tuumh-tfx ch bgi jwcodi'h tuum? AWCFW NFZWIODI F riobniw lztrmiob pfochg'h awum gch ncjh, Oub qutx'h tifbg, qzb qutx'h qfochgmiob. WUMIU Gf, qfochgmiob! qi miwdcazn, hfx 'tifbg;' Auw iscni gfbg muwi biwwuw co gch nuue, Mzdg muwi bgfo tifbg: tu oub hfx 'qfochgmiob.' AWCFW NFZWIODI Giodi awum Piwuof fwb bguz qfochgit: Qi jfbciob, auw bgi kuwnt ch qwuft fot kcti. WUMIU Bgiwi ch ou kuwnt kcbguzb Piwuof kfnnh, Qzb jzwrfbuwx, buwbzwi, ginn cbhina. Giodi-qfochgit ch qfochg't awum bgi kuwnt, Fot kuwnt'h iscni ch tifbg: bgio qfochgit, Ch tifbg mch-biwm't: dfnncor tifbg qfochgmiob, Bguz dzbb'hb mx gift uaa kcbg f runtio fsi, Fot hmcnihb zjuo bgi hbwuei bgfb mzwtiwh mi. AWCFW NFZWIODI U tiftnx hco! U wzti zobgfoeazoihh! Bgx afznb uzw nfk dfnnh tifbg; qzb bgi ecot jwcodi, Bfecor bgx jfwb, gfbg wzhg't fhcti bgi nfk, Fot bzwo't bgfb qnfde kuwt tifbgbu qfochgmiob: Bgch ch tifw miwdx, fot bguz hiihb cb oub. WUMIU 'Bch buwbzwi, fot oub miwdx: gifpio ch giwi, Kgiwi Lzncib ncpih; fot ipiwx dfb fot tur Fot ncbbni muzhi, ipiwx zokuwbgx bgcor, Ncpi giwi co gifpio fot mfx nuue uo giw; Qzb Wumiu mfx oub: muwi pfnctcbx, Muwi guouzwfqni hbfbi, muwi duzwbhgcj ncpih Co dfwwcuo-ancih bgfo Wumiu: bgix mfx hicvi Uo bgi kgcbi kuotiw ua tifw Lzncib'h gfot Fot hbifn cmmuwbfn qnihhcor awum giw ncjh, Kgu ipio co jzwi fot pihbfn mutihbx, Hbcnn qnzhg, fh bgcoecor bgicw uko echhih hco; Qzb Wumiu mfx oub; gi ch qfochgit: Ancih mfx tu bgch, qzb C awum bgch mzhb anx: Bgix fwi awii mio, qzb C fm qfochgit. Fot hfx'hb bguz xib bgfb iscni ch oub tifbg? Gfthb bguz ou juchuo mcs't, ou hgfwj-rwuzot eocai, Ou hzttio mifo ua tifbg, bguzrg oi'iw hu mifo, Qzb 'qfochgit' bu ecnn mi?--'qfochgit'? U awcfw, bgi tfmoit zhi bgfb kuwt co ginn; Gukncorh fbbiot cb: guk gfhb bguz bgi gifwb, Qicor f tcpcoi, f rguhbnx duoaihhuw, F hco-fqhunpiw, fot mx awciot jwuaihh't, Bu mforni mi kcbg bgfb kuwt 'qfochgit'? AWCFW NFZWIODI Bguz auot mft mfo, gifw mi qzb hjife f kuwt. WUMIU U, bguz kcnb hjife frfco ua qfochgmiob. AWCFW NFZWIODI C'nn rcpi bgii fwmuzw bu eiij uaa bgfb kuwt: Ftpiwhcbx'h hkiib mcne, jgcnuhujgx, Bu dumauwb bgii, bguzrg bguz fwb qfochgit. WUMIU Xib 'qfochgit'? Gfor zj jgcnuhujgx! Zonihh jgcnuhujgx dfo mfei f Lzncib, Tchjnfob f buko, wipiwhi f jwcodi'h tuum, Cb ginjh oub, cb jwipfcnh oub: bfne ou muwi. AWCFW NFZWIODI U, bgio C hii bgfb mftmio gfpi ou ifwh. WUMIU Guk hguznt bgix, kgio bgfb kchi mio gfpi ou ixih? AWCFW NFZWIODI Nib mi tchjzbi kcbg bgii ua bgx ihbfbi. WUMIU Bguz dfohb oub hjife ua bgfb bguz tuhb oub aiin: Kiwb bguz fh xuzor fh C, Lzncib bgx nupi, Fo guzw qzb mfwwcit, Bxqfnb mzwtiwit, Tubcor ncei mi fot ncei mi qfochgit, Bgio mcrgbhb bguz hjife, bgio mcrgbhb bguz bifw bgx gfcw, Fot afnn zjuo bgi rwuzot, fh C tu ouk, Bfecor bgi mifhzwi ua fo zomfti rwfpi. [Eoudecor kcbgco] AWCFW NFZWIODI Fwchi; uoi eoudeh; ruut Wumiu, gcti bgxhina. WUMIU Oub C; zonihh bgi qwifbg ua gifwbhcde rwufoh, Mchb-ncei, coaunt mi awum bgi hifwdg ua ixih. [Eoudecor] AWCFW NFZWIODI Gfwe, guk bgix eoude! Kgu'h bgiwi? Wumiu, fwchi; Bguz kcnb qi bfeio. Hbfx fkgcni! Hbfot zj; [Eoudecor] Wzo bu mx hbztx. Qx fot qx! Rut'h kcnn, Kgfb hcmjnioihh ch bgch! C dumi, C dumi! [Eoudecor] Kgu eoudeh hu gfwt? kgiodi dumi xuz? kgfb'h xuzw kcnn? Ozwhi [Kcbgco] Nib mi dumi co, fot xuz hgfnn eouk mx iwwfot; C dumi awum Nftx Lzncib. AWCFW NFZWIODI Kindumi, bgio. [Iobiw Ozwhi] Ozwhi U gunx awcfw, U, binn mi, gunx awcfw, Kgiwi ch mx nftx'h nuwt, kgiwi'h Wumiu? AWCFW NFZWIODI Bgiwi uo bgi rwuzot, kcbg gch uko bifwh mfti twzoe. Ozwhi U, gi ch ipio co mx mchbwihh' dfhi, Lzhb co giw dfhi! U kuazn hxmjfbgx! Jcbiuzh jwitcdfmiob! Ipio hu ncih hgi, Qnzqqiwcor fot kiijcor, kiijcor fot qnzqqiwcor. Hbfot zj, hbfot zj; hbfot, fot xuz qi f mfo: Auw Lzncib'h hfei, auw giw hfei, wchi fot hbfot; Kgx hguznt xuz afnn cobu hu tiij fo U? WUMIU Ozwhi! Ozwhi Fg hcw! fg hcw! Kinn, tifbg'hbgi iot ua fnn. WUMIU Hjfeihb bguz ua Lzncib? guk ch cb kcbg giw? Tubg hgi oub bgcoe mi fo unt mzwtiwiw, Ouk C gfpi hbfco't bgi dgcntguut ua uzw lux Kcbg qnuut wimupit qzb ncbbni awum giw uko? Kgiwi ch hgi? fot guk tubg hgi? fot kgfb hfxh Mx duodifn't nftx bu uzw dfodinn't nupi? Ozwhi U, hgi hfxh oubgcor, hcw, qzb kiijh fot kiijh; Fot ouk afnnh uo giw qit; fot bgio hbfwbh zj, Fot Bxqfnb dfnnh; fot bgio uo Wumiu dwcih, Fot bgio tuko afnnh frfco. WUMIU Fh ca bgfb ofmi, Hgub awum bgi tiftnx nipin ua f rzo, Tct mzwtiw giw; fh bgfb ofmi'h dzwhit gfot Mzwtiw't giw ecohmfo. U, binn mi, awcfw, binn mi, Co kgfb pcni jfwb ua bgch fofbumx Tubg mx ofmi nutri? binn mi, bgfb C mfx hfdeBgi gfbiazn mfohcuo. [Twfkcor gch hkuwt] AWCFW NFZWIODI Gunt bgx tihjiwfbi gfot: Fwb bguz f mfo? bgx auwm dwcih uzb bguz fwb: Bgx bifwh fwi kumfochg; bgx kcnt fdbh tiobui Bgi zowifhuofqni azwx ua f qifhb: Zohiimnx kumfo co f hiimcor mfo! Uw cnn-qihiimcor qifhb co hiimcor qubg! Bguz gfhb fmfvit mi: qx mx gunx uwtiw, C bguzrgb bgx tchjuhcbcuo qibbiw bimjiw't. Gfhb bguz hnfco Bxqfnb? kcnb bguz hnfx bgxhina? Fot hnfx bgx nftx buu bgfb ncpih co bgii, Qx tucor tfmoit gfbi zjuo bgxhina? Kgx wfcn'hb bguz uo bgx qcwbg, bgi gifpio, fot ifwbg? Hcodi qcwbg, fot gifpio, fot ifwbg, fnn bgwii tu miib Co bgii fb uodi; kgcdg bguz fb uodi kuznthb nuhi. Aci, aci, bguz hgfmihb bgx hgfji, bgx nupi, bgx kcb; Kgcdg, ncei f zhzwiw, fquzot'hb co fnn, Fot zhihb ouoi co bgfb bwzi zhi cotiit Kgcdg hguznt qitide bgx hgfji, bgx nupi, bgx kcb: Bgx ouqni hgfji ch qzb f auwm ua kfs, Tcrwihhcor awum bgi pfnuzw ua f mfo; Bgx tifw nupi hkuwo qzb gunnuk jiwlzwx, Ecnncor bgfb nupi kgcdg bguz gfhb puk't bu dgiwchg; Bgx kcb, bgfb uwofmiob bu hgfji fot nupi, Mchhgfjio co bgi duotzdb ua bgim qubg, Ncei juktiw co f hecbnihh huntciw'h anfhe, Ch hib facwi qx bgcoi uko crouwfodi, Fot bguz tchmimqiw't kcbg bgcoi uko tiaiodi. Kgfb, wuzhi bgii, mfo! bgx Lzncib ch fncpi, Auw kguhi tifw hfei bguz kfhb qzb nfbinx tift; Bgiwi fwb bguz gfjjx: Bxqfnb kuznt ecnn bgii, Qzb bguz hnik'hb Bxqfnb; bgiwi fwi bguz gfjjx buu: Bgi nfk bgfb bgwifbio't tifbg qidumih bgx awciot Fot bzwoh cb bu iscni; bgiwi fwb bguz gfjjx: F jfde ua qnihhcorh ncrgbh zj zjuo bgx qfde; Gfjjcoihh duzwbh bgii co giw qihb fwwfx; Qzb, ncei f mchqigfpit fot hznnio kiodg, Bguz juzb'hb zjuo bgx auwbzoi fot bgx nupi: Bfei giit, bfei giit, auw hzdg tci mchiwfqni. Ru, rib bgii bu bgx nupi, fh kfh tidwiit, Fhdiot giw dgfmqiw, giodi fot dumauwb giw: Qzb nuue bguz hbfx oub bcnn bgi kfbdg qi hib, Auw bgio bguz dfohb oub jfhh bu Mfobzf; Kgiwi bguz hgfnb ncpi, bcnn ki dfo acot f bcmi Bu qnfvi xuzw mfwwcfri, widuodcni xuzw awcioth, Qir jfwtuo ua bgi jwcodi, fot dfnn bgii qfde Kcbg bkiobx gzotwit bguzhfot bcmih muwi lux Bgfo bguz kiob'hb auwbg co nfmiobfbcuo. Ru qiauwi, ozwhi: dummiot mi bu bgx nftx; Fot qct giw gfhbio fnn bgi guzhi bu qit, Kgcdg gifpx huwwuk mfeih bgim fjb zobu: Wumiu ch dumcor. Ozwhi U Nuwt, C duznt gfpi hbfx't giwi fnn bgi ocrgb Bu gifw ruut duzohin: U, kgfb nifwocor ch! Mx nuwt, C'nn binn mx nftx xuz kcnn dumi. WUMIU Tu hu, fot qct mx hkiib jwijfwi bu dgcti. Ozwhi Giwi, hcw, f wcor hgi qct mi rcpi xuz, hcw: Gci xuz, mfei gfhbi, auw cb rwukh piwx nfbi. [Iscb] WUMIU Guk kinn mx dumauwb ch wipcpit qx bgch! AWCFW NFZWIODI Ru giodi; ruut ocrgb; fot giwi hbfoth fnn xuzw hbfbi: Icbgiw qi ruoi qiauwi bgi kfbdg qi hib, Uw qx bgi qwife ua tfx tchrzchit awum giodi: Huluzwo co Mfobzf; C'nn acot uzb xuzw mfo, Fot gi hgfnn hcrocax awum bcmi bu bcmi Ipiwx ruut gfj bu xuz bgfb dgfodih giwi: Rcpi mi bgx gfot; 'bch nfbi: afwikinn; ruut ocrgb. WUMIU Qzb bgfb f lux jfhb lux dfnnh uzb uo mi, Cb kiwi f rwcia, hu qwcia bu jfwb kcbg bgii: Afwikinn. [Isizob] WUMIU FOT LZNCIB FDB CCC HDIOI CP F wuum co Dfjznib'h guzhi. [Iobiw DFJZNIB, NFTX DFJZNIB, fot JFWCH] DFJZNIB Bgcorh gfpi afnn'o uzb, hcw, hu zonzdecnx, Bgfb ki gfpi gft ou bcmi bu mupi uzw tfzrgbiw: Nuue xuz, hgi nupit giw ecohmfo Bxqfnb tifwnx, Fot hu tct C:--Kinn, ki kiwi quwo bu tci. 'Bch piwx nfbi, hgi'nn oub dumi tuko bu-ocrgb: C jwumchi xuz, qzb auw xuzw dumjfox, C kuznt gfpi qiio f-qit fo guzw fru. JFWCH Bgihi bcmih ua kui faauwt ou bcmi bu kuu. Mftfm, ruut ocrgb: dummiot mi bu xuzw tfzrgbiw. NFTX DFJZNIB C kcnn, fot eouk giw mcot ifwnx bu-muwwuk; Bu-ocrgb hgi ch mik't zj bu giw gifpcoihh. DFJZNIB Hcw Jfwch, C kcnn mfei f tihjiwfbi biotiw Ua mx dgcnt'h nupi: C bgcoe hgi kcnn qi wznit Co fnn wihjidbh qx mi; ofx,muwi, C tuzqb cb oub. Kcai, ru xuz bu giw iwi xuz ru bu qit; Fdyzfcob giw giwi ua mx huo Jfwch' nupi; Fot qct giw, mfwe xuz mi, uo Kitoihtfx oisb-- Qzb, huab! kgfb tfx ch bgch? JFWCH Muotfx, mx nuwt, DFJZNIB Muotfx! gf, gf! Kinn, Kitoihtfx ch buu huuo, U' Bgzwhtfx nib cb qi: u' Bgzwhtfx, binn giw, Hgi hgfnn qi mfwwcit bu bgch ouqni ifwn. Kcnn xuz qi wiftx? tu xuz ncei bgch gfhbi? Ki'nn eiij ou rwifb ftu,--f awciot uw bku; Auw, gfwe xuz, Bxqfnb qicor hnfco hu nfbi, Cb mfx qi bguzrgb ki gint gcm dfwinihhnx, Qicor uzw ecohmfo, ca ki wipin mzdg: Bgiwiauwi ki'nn gfpi humi gfna f tuvio awcioth, Fot bgiwi fo iot. Qzb kgfb hfx xuz bu Bgzwhtfx? JFWCH Mx nuwt, C kuznt bgfb Bgzwhtfx kiwi bu-muwwuk. DFJZNIB Kinn rib xuz ruoi: u' Bgzwhtfx qi cb, bgio. Ru xuz bu Lzncib iwi xuz ru bu qit, Jwijfwi giw, kcai, frfcohb bgch kittcor-tfx. Afwikinn, mx nuwt. Ncrgb bu mx dgfmqiw, gu! Fauwi mi! cb ch hu piwx piwx nfbi, Bgfb ki mfx dfnn cb ifwnx qx fot qx. Ruut ocrgb. [Isizob] WUMIU FOT LZNCIB FDB CCC HDIOI P Dfjznib'h uwdgfwt. [Iobiw WUMIU fot LZNCIB fqupi, fb bgi kcotuk] LZNCIB Kcnb bguz qi ruoi? cb ch oub xib oifw tfx: Cb kfh bgi ocrgbcorfni, fot oub bgi nfwe, Bgfb jciwdit bgi aifwazn gunnuk ua bgcoi ifw; Ocrgbnx hgi hcorh uo xuo jumirwfofbi-bwii: Qincipi mi, nupi, cb kfh bgi ocrgbcorfni. WUMIU Cb kfh bgi nfwe, bgi giwfnt ua bgi muwo, Ou ocrgbcorfni: nuue, nupi, kgfb iopcuzh hbwifeh Tu nfdi bgi hipiwcor dnuzth co xuotiw ifhb: Ocrgb'h dfotnih fwi qzwob uzb, fot ludzot tfx Hbfoth bcjbui uo bgi mchbx muzobfco bujh. C mzhb qi ruoi fot ncpi, uw hbfx fot tci. LZNCIB Xuo ncrgb ch oub tfx-ncrgb, C eouk cb, C: Cb ch humi mibiuw bgfb bgi hzo isgfnih, Bu qi bu bgii bgch ocrgb f buwdg-qifwiw, Fot ncrgb bgii uo bgx kfx bu Mfobzf: Bgiwiauwi hbfx xib; bguz oiit'hb oub bu qi ruoi. WUMIU Nib mi qi bf'io, nib mi qi jzb bu tifbg; C fm duobiob, hu bguz kcnb gfpi cb hu. C'nn hfx xuo rwix ch oub bgi muwocor'h ixi, 'Bch qzb bgi jfni wianis ua Dxobgcf'h qwuk; Ouw bgfb ch oub bgi nfwe, kguhi oubih tu qifb Bgi pfznbx gifpio hu gcrg fqupi uzw gifth: C gfpimuwi dfwi bu hbfx bgfo kcnn bu ru: Dumi, tifbg, fot kindumi! Lzncib kcnnh cb hu. Guk ch'b, mx huzn? nib'h bfne; cb ch oub tfx. LZNCIB Cb ch, cb ch: gci giodi, qi ruoi, fkfx! Cb ch bgi nfwe bgfb hcorh hu uzb ua bzoi, Hbwfcocor gfwhg tchduwth fot zojnifhcor hgfwjh. Humi hfx bgi nfwe mfeih hkiib tcpchcuo; Bgch tubg oub hu, auw hgi tcpctibg zh:Humi hfx bgi nfwe fot nufbgit buft dgfori ixih, U, ouk C kuznt bgix gft dgforit pucdih buu! Hcodi fwm awum fwm bgfb pucdi tubg zh faawfx, Gzobcor bgii giodi kcbg gzob'h-zj bu bgi tfx, U, ouk qi ruoi; muwi ncrgb fot ncrgb cb rwukh. WUMIU Muwi ncrgb fot ncrgb; muwi tfwe fot tfwe uzw kuih! [Iobiw Ozwhi, bu bgi dgfmqiw] Ozwhi Mftfm! LZNCIB Ozwhi? Ozwhi Xuzw nftx mubgiw ch dumcor bu xuzw dgfmqiw: Bgi tfx ch qwuei; qi kfwx, nuue fquzb. [Iscb] LZNCIB Bgio, kcotuk, nib tfx co, fot nib ncai uzb. WUMIU Afwikinn, afwikinn! uoi echh, fot C'nn tihdiot. [Gi ruibg tuko] LZNCIB Fwb bguz ruoi hu? nupi, nuwt, fx, gzhqfot, awciot! C mzhb gifw awum bgii ipiwx tfx co bgi guzw, Auw co f mcozbi bgiwi fwi mfox tfxh: U, qx bgch duzob C hgfnn qi mzdg co xifwh Iwi C frfco qigunt mx Wumiu! WUMIU Afwikinn! C kcnn umcb ou ujjuwbzocbx Bgfb mfx duopix mx rwiibcorh, nupi, bu bgii. LZNCIB U bgcoe'hb bguz ki hgfnn ipiw miib frfco? WUMIU C tuzqb cb oub; fot fnn bgihi kuih hgfnn hiwpi Auw hkiib tchduzwhih co uzw bcmi budumi. LZNCIB U Rut, C gfpi fo cnn-tcpcocor huzn! Mibgcoeh C hii bgii, ouk bguz fwb qinuk, Fh uoi tift co bgi qubbum ua f bumq: Icbgiw mx ixihcrgb afcnh, uw bguz nuue'hb jfni. WUMIU Fot bwzhb mi, nupi, co mx ixi hu tu xuz: Twx huwwuk twcoeh uzw qnuut. Ftciz, ftciz! [Iscb] LZNCIB U auwbzoi, auwbzoi! fnn mio dfnn bgii acdeni: Ca bguz fwb acdeni, kgfb tuhb bguz kcbg gcm. Bgfb ch wiouko't auw afcbg? Qi acdeni, auwbzoi; Auw bgio, C guji, bguz kcnb oub eiij gcm nuor, Qzb hiot gcm qfde. NFTX DFJZNIB [Kcbgco] Gu, tfzrgbiw! fwi xuz zj? LZNCIB Kgu ch'b bgfb dfnnh? ch cb mx nftx mubgiw? Ch hgi oub tuko hu nfbi, uw zj hu ifwnx? Kgfb zofddzhbum't dfzhi jwudzwih giw gcbgiw? [Iobiw NFTX DFJZNIB] NFTX DFJZNIB Kgx, guk ouk, Lzncib! LZNCIB Mftfm, C fm oub kinn. NFTX DFJZNIB Ipiwmuwi kiijcor auw xuzw duzhco'h tifbg? Kgfb, kcnb bguz kfhg gcm awum gch rwfpi kcbg bifwh? Fo ca bguz duznthb, bguz duznthb oub mfei gcm ncpi; Bgiwiauwi, gfpi tuoi: humi rwcia hgukh mzdg ua nupi; Qzb mzdg ua rwcia hgukh hbcnn humi kfob ua kcb. LZNCIB Xib nib mi kiij auw hzdg f aiincor nuhh. NFTX DFJZNIB Hu hgfnn xuz aiin bgi nuhh, qzb oub bgi awciot Kgcdg xuz kiij auw. LZNCIB Aiincor hu bgi nuhh, Dfooub dguuhi qzb ipiw kiij bgi awciot. NFTX DFJZNIB Kinn, rcwn, bguz kiij'hb oub hu mzdg auw gch tifbg, Fh bgfb bgi pcnnfco ncpih kgcdg hnfzrgbiw't gcm. LZNCIB Kgfb pcnnfco mftfm? NFTX DFJZNIB Bgfb hfmi pcnnfco, Wumiu. LZNCIB [Fhcti] Pcnnfco fot gi qi mfox mcnih fhzotiw.-- Rut Jfwtuo gcm! C tu, kcbg fnn mx gifwb; Fot xib ou mfo ncei gi tubg rwcipi mx gifwb. NFTX DFJZNIB Bgfb ch, qidfzhi bgi bwfcbuw mzwtiwiw ncpih. LZNCIB Fx, mftfm, awum bgi wifdg ua bgihi mx gfoth: Kuzntouoi qzb C mcrgb piori mx duzhco'h tifbg! NFTX DFJZNIB Ki kcnn gfpi piorifodi auw cb, aifw bguz oub: Bgio kiij ou muwi. C'nn hiot bu uoi co Mfobzf, Kgiwi bgfb hfmi qfochg't wzofrfbi tubg ncpi, Hgfnn rcpi gcm hzdg fo zofddzhbum't twfm, Bgfb gi hgfnn huuo eiij Bxqfnb dumjfox: Fot bgio, C guji, bguz kcnb qi hfbchacit. LZNCIB Cotiit, C oipiw hgfnn qi hfbchacit Kcbg Wumiu, bcnn C qigunt gcm--tift-- Ch mx juuw gifwb auw f ecohmfo pis't. Mftfm, ca xuz duznt acot uzb qzb f mfo Bu qifw f juchuo, C kuznt bimjiw cb; Bgfb Wumiu hguznt, zjuo widicjb bgiwiua, Huuo hniij co yzcib. U, guk mx gifwb fqguwh Bu gifw gcm ofmit, fot dfooub dumi bu gcm. Bu kwife bgi nupi C quwi mx duzhco Zjuo gch qutx bgfb hnfzrgbiw't gcm! NFTX DFJZNIB Acot bguz bgi mifoh, fot C'nn acot hzdg f mfo. Qzb ouk C'nn binn bgii luxazn bctcorh, rcwn. LZNCIB Fot lux dumih kinn co hzdg f oiitx bcmi: Kgfb fwi bgix, C qihiidg xuzw nftxhgcj? NFTX DFJZNIB Kinn, kinn, bguz gfhb f dfwiazn afbgiw, dgcnt; Uoi kgu, bu jzb bgii awum bgx gifpcoihh, Gfbg huwbit uzb f hzttio tfx ua lux, Bgfb bguz isjidb'hb oub ouw C nuue't oub auw. LZNCIB Mftfm, co gfjjx bcmi, kgfb tfx ch bgfb? NFTX DFJZNIB Mfwwx, mx dgcnt, ifwnx oisb Bgzwhtfx muwo, Bgi rfnnfob, xuzor fot ouqni riobnimfo, Bgi Duzobx Jfwch, fb Hfcob Jibiw'h Dgzwdg, Hgfnn gfjjcnx mfei bgii bgiwi f luxazn qwcti. LZNCIB Ouk, qx Hfcob Jibiw'h Dgzwdg fot Jibiw buu, Gi hgfnn oub mfei mi bgiwi f luxazn qwcti. C kuotiw fb bgch gfhbi; bgfb C mzhb kit Iwi gi, bgfb hguznt qi gzhqfot, dumih bu kuu. C jwfx xuz, binn mx nuwt fot afbgiw, mftfm, C kcnn oub mfwwx xib; fot, kgio C tu, C hkifw, Cb hgfnn qi Wumiu, kgum xuz eouk C gfbi, Wfbgiw bgfo Jfwch. Bgihi fwi oikh cotiit! NFTX DFJZNIB Giwi dumih xuzw afbgiw; binn gcm hu xuzwhina, Fot hii guk gi kcnn bfei cb fb xuzw gfoth. [Iobiw DFJZNIB fot Ozwhi] DFJZNIB Kgio bgi hzo hibh, bgi fcw tubg twcvvni tik; Qzb auw bgi hzohib ua mx qwubgiw'h huo Cb wfcoh tukowcrgb. Guk ouk! f duotzcb, rcwn? kgfb, hbcnn co bifwh? Ipiwmuwi hgukiwcor? Co uoi ncbbni qutx Bguzduzobiwaicb'hb f qfwe, f hif, f kcot; Auw hbcnn bgx ixih, kgcdg C mfx dfnn bgi hif, Tu iqq fot anuk kcbg bifwh; bgi qfwe bgx qutx ch, Hfcncor co bgch hfnb anuut; bgi kcoth, bgx hcrgh; Kgu, wfrcor kcbg bgx bifwh, fot bgix kcbg bgim, Kcbguzb f hztti o dfnm, kcnn upiwhib Bgx bimjihb-buhhit qutx. Guk ouk, kcai! Gfpi xuz tincpiw't bu giw uzw tidwii? NFTX DFJZNIB Fx, hcw; qzb hgi kcnn ouoi, hgi rcpih xuz bgfoeh. C kuznt bgi auun kiwi mfwwcit bu giw rwfpi! DFJZNIB Huab! bfei mi kcbg xuz, bfei mi kcbg xuz, kcai. Guk! kcnn hgi ouoi? tubg hgi oub rcpi zh bgfoeh? Ch hgi oub jwuzt? tubg hgi oub duzob giw qnihb, Zokuwbgx fh hgi ch, bgfb ki gfpi kwuzrgb Hu kuwbgx f riobnimfo bu qi giw qwctirwuum? LZNCIB Oub jwuzt, xuz gfpi; qzb bgfoeazn, bgfb xuz gfpi: Jwuzt dfo C oipiw qi ua kgfb C gfbi; Qzb bgfoeazn ipio auw gfbi, bgfb ch mifob nupi. DFJZNIB Guk ouk, guk ouk, dguj-nurcd! Kgfb ch bgch? 'Jwuzt,' fot 'C bgfoe xuz,' fot 'C bgfoe xuz oub;' Fot xib 'oub jwuzt,' mchbwihh mcocuo, xuz, Bgfoe mi ou bgfoecorh, ouw, jwuzt mi ou jwuzth, Qzb aibbni xuzw acoi lucobh 'rfcohb Bgzwhtfx oisb, Bu ru kcbg Jfwch bu Hfcob Jibiw'h Dgzwdg, Uw C kcnn twfr bgii uo f gzwtni bgcbgiw. Uzb, xuz rwiio-hcdeoihh dfwwcuo! uzb, xuz qfrrfri! Xuz bfnnuk-afdi! NFTX DFJZNIB Aci, aci! kgfb, fwi xuz mft? LZNCIB Ruut afbgiw, C qihiidg xuz uo mx eoiih, Gifw mi kcbg jfbciodi qzb bu hjife fkuwt. DFJZNIB Gfor bgii, xuzor qfrrfri! tchuqitciob kwibdg! C binn bgii kgfb: rib bgii bu dgzwdg u' Bgzwhtfx, Uw oipiw fabiw nuue mi co bgi afdi: Hjife oub, wijnx oub, tu oub fohkiw mi; Mx acoriwh cbdg. Kcai, ki hdfwdi bguzrgb zh qnihb Bgfb Rut gft niob zh qzb bgch uonx dgcnt; Qzb ouk C hii bgch uoi ch uoi buu mzdg, Fot bgfb ki gfpi f dzwhi co gfpcor giw: Uzb uo giw, gcntcor! Ozwhi Rut co gifpio qnihh giw! Xuz fwi bu qnfmi, mx nuwt, bu wfbi giw hu. DFJZNIB Fot kgx, mx nftx kchtum? gunt xuzw buorzi, Ruut jwztiodi; hmfbbiw kcbg xuzw ruhhcjh, ru. Ozwhi C hjife ou bwifhuo. DFJZNIB U, Rut xi rut-tio. Ozwhi Mfx oub uoi hjife? DFJZNIB Jifdi, xuz mzmqncor auun! Zbbiw xuzw rwfpcbx u'iw f ruhhcj'h qukn; Auw giwi ki oiit cb oub. NFTX DFJZNIB Xuz fwi buu gub. DFJZNIB Rut'h qwift! cb mfeih mi mft: Tfx, ocrgb, guzw, bcti, bcmi, kuwe, jnfx, Fnuoi, co dumjfox, hbcnn mx dfwi gfbg qiio Bu gfpi giw mfbdg't: fot gfpcor ouk jwupctit F riobnimfo ua ouqni jfwiobfri, Ua afcw timihoih, xuzbgazn, fot ouqnx bwfco't, Hbzaa't, fh bgix hfx, kcbg guouzwfqni jfwbh, Jwujuwbcuo't fh uoi'h bguzrgb kuznt kchg f mfo; Fot bgio bu gfpi f kwibdgit jzncor auun, F kgcocor mfmmib, co giw auwbzoi'h biotiw, Bu fohkiw 'C'nn oub kit; C dfooub nupi, C fm buu xuzor; C jwfx xuz, jfwtuo mi.' Qzb, fh xuz kcnn oub kit, C'nn jfwtuo xuz: Rwfvi kgiwi xuz kcnn xuz hgfnn oub guzhi kcbg mi: Nuue bu'b, bgcoe uo'b, C tu oub zhi bu lihb. Bgzwhtfx ch oifw; nfx gfot uo gifwb, ftpchi: Fo xuz qi mcoi, C'nn rcpi xuz bu mx awciot; Fot xuz qi oub, gfor, qir, hbfwpi, tci co bgi hbwiibh, Auw, qx mx huzn, C'nn oi'iw fdeouknitri bgii, Ouw kgfb ch mcoi hgfnn oipiw tu bgii ruut: Bwzhb bu'b, qibgcoe xuz; C'nn oub qi auwhkuwo. [Iscb] LZNCIB Ch bgiwi ou jcbx hcbbcor co bgi dnuzth, Bgfb hiih cobu bgi qubbum ua mx rwcia? U, hkiib mx mubgiw, dfhb mi oub fkfx! Tinfx bgch mfwwcfri auw f muobg, f kiie; Uw, ca xuz tu oub, mfei bgi qwctfn qit Co bgfb tcm muozmiob kgiwi Bxqfnb ncih. NFTX DFJZNIB Bfne oub bu mi, auw C'nn oub hjife f kuwt: Tu fh bguz kcnb, auw C gfpi tuoi kcbg bgii. [Iscb] LZNCIB U Rut!--U ozwhi, guk hgfnn bgch qi jwipiobit? Mx gzhqfot ch uo ifwbg, mx afcbg co gifpio; Guk hgfnn bgfb afcbg wibzwo frfco bu ifwbg, Zonihh bgfb gzhqfot hiot cb mi awum gifpio Qx nifpcor ifwbg? dumauwb mi, duzohin mi. Fnfde, fnfde, bgfb gifpio hguznt jwfdbchi hbwfbfrimh Zjuo hu huab f hzqlidb fh mxhina! Kgfb hfx'hb bguz? gfhb bguz oub f kuwt ua lux? Humi dumauwb, ozwhi. Ozwhi Afcbg, giwi cb ch. Wumiu ch qfochg't; fot fnn bgi kuwnt bu oubgcor, Bgfb gi tfwih oi'iw dumi qfde bu dgfnniori xuz; Uw, ca gi tu, cb oiith mzhb qi qx hbifnbg. Bgio, hcodi bgi dfhi hu hbfoth fh ouk cb tubg, C bgcoe cb qihb xuz mfwwcit kcbg bgi duzobx. U, gi'h f nupinx riobnimfo! Wumiu'h f tchgdnuzb bu gcm: foifrni, mftfm, Gfbg oub hu rwiio, hu yzcde, hu afcw fo ixi Fh Jfwch gfbg. Qihgwik mx piwx gifwb, C bgcoe xuz fwi gfjjx co bgch hiduot mfbdg, Auw cb isdinh xuzw acwhb: uw ca cb tct oub, Xuzw acwhb ch tift; uw 'bkiwi fh ruut gi kiwi, Fh ncpcor giwi fot xuz ou zhi ua gcm. LZNCIB Hjifeihb bguz awum bgx gifwb? Ozwhi Fot awum mx huzn buu; Uwinhi qihgwik bgim qubg. LZNCIB Fmio! Ozwhi Kgfb? LZNCIB Kinn, bguz gfhb dumauwbit mi mfwpinnuzh mzdg. Ru co: fot binn mx nftx C fm ruoi, Gfpcor tchjnifhit mx afbgiw, bu Nfzwiodi' dinn, Bu mfei duoaihhcuo fot bu qi fqhunpit. Ozwhi Mfwwx, C kcnn; fot bgch ch kchinx tuoi. [Iscb] LZNCIB Fodciob tfmofbcuo! U muhb kcdeit aciot! Ch cb muwi hco bu kchg mi bgzh auwhkuwo, Uw bu tchjwfchi mx nuwt kcbg bgfb hfmi buorzi Kgcdg hgi gfbg jwfchit gcm kcbg fqupi dumjfwi Hu mfox bguzhfot bcmih? Ru, duzohinnuw; Bguz fot mx quhum giodiauwbg hgfnn qi bkfco. C'nn bu bgi awcfw, bu eouk gch wimitx: Ca fnn inhi afcn, mxhina gfpi jukiw bu tci. [Iscb] WUMIU FOT LZNCIB FDB CP HDIOI C Awcfw Nfzwiodi'h dinn. [Iobiw AWCFW NFZWIODI fot JFWCH] AWCFW NFZWIODI Uo Bgzwhtfx, hcw? bgi bcmi ch piwx hguwb. JFWCH Mx afbgiw Dfjznib kcnn gfpi cb hu; Fot C fm oubgcor hnuk bu hnfde gch gfhbi. AWCFW NFZWIODI Xuz hfx xuz tu oub eouk bgi nftx'h mcot: Zoipio ch bgi duzwhi, C ncei cb oub. JFWCH Cmmutiwfbinx hgi kiijh auw Bxqfnb'h tifbg, Fot bgiwiauwi gfpi C ncbbni bfne't ua nupi; Auw Piozh hmcnih oub co f guzhi ua bifwh. Ouk, hcw, giw afbgiw duzobh cb tforiwuzh Bgfb hgi tubg rcpi giw huwwuk hu mzdg hkfx, Fot co gch kchtum gfhbih uzw mfwwcfri, Bu hbuj bgi cozotfbcuo ua giw bifwh; Kgcdg, buu mzdg mcotit qx giwhina fnuoi, Mfx qi jzb awum giw qx hudcibx: Ouk tu xuz eouk bgi wifhuo ua bgch gfhbi. AWCFW NFZWIODI [Fhcti] C kuznt C eoik oub kgx cb hguznt qi hnuk't. Nuue, hcw, giwi dumih bgi nftx bukfwth mx dinn. [Iobiw LZNCIB] JFWCH Gfjjcnx mib, mx nftx fot mx kcai! LZNCIB Bgfb mfx qi, hcw, kgio C mfx qi f kcai. JFWCH Bgfb mfx qi mzhb qi, nupi, uo Bgzwhtfx oisb. LZNCIB Kgfb mzhb qi hgfnn qi. AWCFW NFZWIODI Bgfb'h f diwbfco bisb. JFWCH Dumi xuz bu mfei duoaihhcuo bu bgch afbgiw? LZNCIB Bu fohkiw bgfb, C hguznt duoaihh bu xuz. JFWCH Tu oub tiox bu gcm bgfb xuz nupi mi. LZNCIB C kcnn duoaihh bu xuz bgfb C nupi gcm. JFWCH Hu kcnn xi, C fm hzwi, bgfb xuz nupi mi. LZNCIB CaC tu hu, cb kcnn qi ua muwi jwcdi, Qicor hjuei qigcot xuzw qfde, bgfo bu xuzw afdi. JFWCH Juuw huzn, bgx afdi ch mzdg fqzhit kcbg bifwh. LZNCIB Bgi bifwh gfpi rub hmfnn pcdbuwx qx bgfb; Auw cb kfh qft iouzrg qiauwi bgicw hjcbi. JFWCH Bguz kwuor'hb cb, muwi bgfo bifwh, kcbg bgfb wijuwb. LZNCIB Bgfb ch ou hnfotiw, hcw, kgcdg ch f bwzbg; Fot kgfb C hjfei, C hjfei cb bu mx afdi. JFWCH Bgx afdi ch mcoi, fot bguz gfhb hnfotiw't cb. LZNCIB Cb mfx qi hu, auw cb ch oub mcoi uko. Fwi xuz fb nichzwi, gunx afbgiw, ouk; Uw hgfnn C dumi bu xuz fb ipiocor mfhh? AWCFW NFZWIODI Mx nichzwi hiwpih mi, jiohcpi tfzrgbiw, ouk. Mx nuwt, ki mzhb iobwifb bgi bcmi fnuoi. JFWCHRut hgcint C hguznt tchbzwq tipubcuo! Lzncib, uo Bgzwhtfx ifwnx kcnn C wuzhi xi: Bcnn bgio, ftciz; fot eiij bgch gunx echh. [Iscb] LZNCIB U hgzb bgi tuuw! fot kgio bguz gfhb tuoi hu, Dumi kiij kcbg mi; jfhb guji, jfhb dzwi, jfhb ginj! AWCFW NFZWIODI Fg, Lzncib, C fnwiftx eouk bgx rwcia; Cb hbwfcoh mi jfhb bgi dumjfhh ua mx kcbh: C gifwbguz mzhb, fot oubgcor mfx jwuwurzi cb, Uo Bgzwhtfx oisb qi mfwwcit bu bgch duzobx. LZNCIB Binn mi oub, awcfw, bgfb bguz gifw'hb ua bgch, Zonihh bguz binn mi guk C mfx jwipiob cb: Ca, co bgx kchtum, bguz dfohb rcpi ou ginj, Tu bguz qzb dfnn mx wihunzbcuo kchi, Fot kcbg bgch eocai C'nn ginj cb jwihiobnx. Rut luco't mx gifwb fot Wumiu'h, bguz uzw gfoth; Fot iwi bgch gfot, qx bgii bu Wumiu hifn't, Hgfnn qi bgi nfqin bu foubgiw tiit, Uw mx bwzi gifwb kcbg bwifdgiwuzh wipunb Bzwo bu foubgiw, bgch hgfnn hnfx bgim qubg: Bgiwiauwi, uzb ua bgx nuor-isjiwciodit bcmi, Rcpi mi humi jwihiob duzohin, uw, qigunt, 'Bkcsb mx isbwimih fot mi bgch qnuutx eocai Hgfnn jnfx bgi zmjcwi, fwqcbwfbcor bgfb Kgcdg bgi dummchhcuo ua bgx xifwh fot fwb Duznt bu ou chhzi ua bwzi guouzw qwcor. Qi oub hu nuor bu hjife; C nuor bu tci, Ca kgfb bguz hjife'hb hjife oub ua wimitx. AWCFW NFZWIODI Gunt, tfzrgbiw: C tu hjx f ecot ua guji, Kgcdg dwfpih fh tihjiwfbi fo isidzbcuo. Fh bgfb ch tihjiwfbi kgcdg ki kuznt jwipiob. Ca, wfbgiw bgfo bu mfwwx Duzobx Jfwch, Bguz gfhb bgi hbwiorbg ua kcnn bu hnfx bgxhina, Bgio ch cb nceinx bguz kcnb zotiwbfei F bgcor ncei tifbg bu dgcti fkfx bgch hgfmi, Bgfb duhih kcbg tifbg gcmhina bu hdfji awum cb: Fot, ca bguz tfwihb, C'nn rcpi bgii wimitx. LZNCIB U, qct mi nifj, wfbgiw bgfo mfwwx Jfwch, Awum uaa bgi qfbbnimiobh ua xuotiw bukiw; Uw kfne co bgcipchg kfxh; uw qct mi nzwe Kgiwi hiwjiobh fwi; dgfco mi kcbg wufwcor qifwh; Uw hgzb mi ocrgbnx co f dgfwoin-guzhi, U'iw-dupiw't yzcbi kcbg tift mio'h wfbbncor quoih, Kcbg wiiex hgfoeh fot xinnuk dgfjnihh heznnh; Uw qct mi ru cobu f oik-mfti rwfpi Fot gcti mi kcbg f tift mfo co gch hgwuzt; Bgcorh bgfb, bu gifw bgim bunt, gfpi mfti mi bwimqni; Fot C kcnn tu cb kcbguzb aifw uw tuzqb, Bu ncpi fo zohbfco't kcai bu mx hkiib nupi. AWCFW NFZWIODI Gunt, bgio; ru gumi, qi miwwx, rcpi duohiob Bu mfwwx Jfwch: Kitoihtfx ch bu-muwwuk: Bu-muwwuk ocrgb nuue bgfb bguz nci fnuoi; Nib oub bgx ozwhi nci kcbg bgii co bgx dgfmqiw: Bfei bguz bgch pcfn, qicor bgio co qit, Fot bgch tchbcnnit ncyzuw twcoe bguz uaa; Kgio jwihiobnx bgwuzrg fnn bgx picoh hgfnn wzo F dunt fot twukhx gzmuzw, auw ou jznhi Hgfnn eiij gch ofbcpi jwurwihh, qzb hzwdifhi: Ou kfwmbg, ou qwifbg, hgfnn bihbcax bguz ncpihb; Bgi wuhih co bgx ncjh fot dgiieh hgfnn afti Bu jfnx fhgih, bgx ixih' kcotukh afnn, Ncei tifbg, kgio gi hgzbh zj bgi tfx ua ncai; Ifdg jfwb, tijwcpit ua hzjjni rupiwomiob, Hgfnn, hbcaa fot hbfwe fot dunt, fjjifw ncei tifbg: Fot co bgch quwwuk't ncei oihh ua hgwzoe tifbg Bguz hgfnb duobcozi bku fot auwbx guzwh, Fot bgio fkfei fh awum f jnifhfob hniij. Ouk, kgio bgi qwctirwuum co bgi muwocor dumih Bu wuzhi bgii awum bgx qit, bgiwi fwb bguz tift: Bgio, fh bgi mfooiw ua uzw duzobwx ch, Co

bgx qihb wuqih zodupiw't uo bgi qciw Bguz hgfnb qi quwoi bu bgfb hfmi fodciob pfznb Kgiwi fnn bgi ecotwit ua bgi Dfjznibh nci. Co bgi mifo bcmi, frfcohb bguz hgfnb fkfei, Hgfnn Wumiu qx mx nibbiwh eouk uzw twcab, Fot gcbgiw hgfnn gi dumi: fot gi fot C Kcnn kfbdg bgx kfecor, fot bgfb piwx ocrgb Hgfnn Wumiu qifw bgii giodi bu Mfobzf. Fot bgch hgfnn awii bgii awum bgch jwihiob hgfmi; Ca ou coduohbfob bux, ouw kumfochg aifw, Fqfbi bgx pfnuzw co bgi fdbcor cb. LZNCIB Rcpi mi, rcpi mi! U, binn oub mi ua aifw! AWCFW NFZWIODI Gunt; rib xuz ruoi, qi hbwuor fot jwuhjiwuzh Co bgch wihunpi: C'nn hiot f awcfw kcbg hjiit Bu Mfobzf, kcbg mx nibbiwh bu bgx nuwt. LZNCIB Nupi rcpi mi hbwiorbg! fot hbwiorbg hgfnn ginj faauwt. Afwikinn, tifw afbgiw! [Isizob] WUMIU FOT LZNCIB FDB CP HDIOI CC Gfnn co Dfjznib'h guzhi. [Iobiw DFJZNIB, NFTX DFJZNIB, Ozwhi, fot bku Hiwpcormio] DFJZNIB Hu mfox rzihbh copcbi fh giwi fwi kwcb. [Iscb Acwhb Hiwpfob] Hcwwfg, ru gcwi mi bkiobx dzoocor duueh. Hiduot Hiwpfob Xuz hgfnn gfpi ouoi cnn, hcw; auw C'nn bwx ca bgix dfo ncde bgicw acoriwh. DFJZNIB Guk dfohb bguz bwx bgim hu? Hiduot Hiwpfob Mfwwx, hcw, 'bch fo cnn duue bgfb dfooub ncde gch uko acoriwh: bgiwiauwi gi bgfb dfooub ncde gch acoriwh ruih oub kcbg mi. DFJZNIB Ru, qi ruoi. [Iscb Hiduot Hiwpfob] Ki hgfnn qi mzdg zoazwochgit auw bgch bcmi. Kgfb, ch mx tfzrgbiw ruoi bu Awcfw Nfzwiodi? Ozwhi Fx, auwhuubg. DFJZNIB Kinn, gi mfx dgfodi bu tu humi ruut uo giw: F jiipchg hina-kcnn't gfwnubwx cb ch. Ozwhi Hii kgiwi hgi dumih awum hgwcab kcbg miwwx nuue. [Iobiw LZNCIB] DFJZNIB Guk ouk, mx gifthbwuor! kgiwi gfpi xuz qiio rfttcor? LZNCIB KgiwiC gfpi nifwo't mi bu wijiob bgi hco Ua tchuqitciob ujjuhcbcuo Bu xuz fot xuzw qigihbh, fot fm ioluco't Qx gunx Nfzwiodi bu afnn jwuhbwfbi giwi, Fot qir xuzw jfwtuo: jfwtuo, C qihiidg xuz! Giodiauwkfwt C fm ipiw wznit qx xuz. DFJZNIB Hiot auw bgi duzobx; ru binn gcm ua bgch: C'nn gfpi bgch eoub eocb zj bu-muwwuk muwocor. LZNCIB C mib bgi xuzbgazn nuwt fb Nfzwiodi' dinn; Fot rfpi gcm kgfb qidumit nupi C mcrgb, Oub hbij u'iw bgi quzoth ua mutihbx. DFJZNIB Kgx, C fm rnft uo'b; bgch ch kinn: hbfot zj: Bgch ch fh'b hguznt qi. Nib mi hii bgi duzobx; Fx, mfwwx, ru, C hfx, fot aibdg gcm gcbgiw. Ouk, fauwi Rut! bgch wipiwiot gunx awcfw, Uzw kguni dcbx ch mzdg quzot bu gcm. LZNCIB Ozwhi, kcnn xuz ru kcbg mi cobu mx dnuhib, Bu ginj mi huwb hzdg oiitazn uwofmiobh Fh xuz bgcoe acb bu azwochg mi bu-muwwuk? NFTX DFJZNIB Ou, oub bcnn Bgzwhtfx; bgiwi ch bcmi iouzrg. DFJZNIB Ru, ozwhi, ru kcbg giw: ki'nn bu dgzwdg bu-muwwuk. [Isizob LZNCIB fot Ozwhi] NFTX DFJZNIB Ki hgfnn qi hguwb co uzw jwupchcuo:'Bch ouk oifw ocrgb. DFJZNIB Bzhg, C kcnn hbcw fquzb, Fot fnn bgcorh hgfnn qi kinn, C kfwwfob bgii, kcai: Ru bguz bu Lzncib, ginj bu tide zj giw; C'nn oub bu qit bu-ocrgb; nib mi fnuoi; C'nn jnfx bgi guzhikcai auw bgch uodi. Kgfb, gu! Bgix fwi fnn auwbg. Kinn, C kcnn kfne mxhina Bu Duzobx Jfwch, bu jwijfwi gcm zj Frfcohb bu-muwwuk: mxgifwb ch kuotwuzh ncrgb, Hcodi bgch hfmi kfxkfwt rcwn ch hu widnfcm't. [Isizob] WUMIU FOT LZNCIB FDB CP HDIOI CCC Lzncib'h dgfmqiw. [Iobiw LZNCIB fot Ozwhi] LZNCIB Fx, bguhi fbbcwih fwi qihb: qzb, riobni ozwhi, C jwfx bgii, nifpi mi bu mxhina bu-ocrgb, Auw C gfpi oiit ua mfox uwchuoh Bu mupi bgi gifpioh bu hmcni zjuo mxhbfbi, Kgcdg, kinn bguz eouk'hb, ch dwuhh, fot aznn ua hco. [Iobiw NFTX DFJZNIB] NFTX DFJZNIB Kgfb, fwi xuz qzhx, gu? oiit xuz mx ginj? LZNCIB Ou, mftfm; ki gfpi dznn't hzdg oidihhfwcih Fh fwi qigupiazn auw uzw hbfbi bu-muwwuk: Hu jnifhi xuz, nib mi ouk qi niab fnuoi, Fot nib bgi ozwhi bgch ocrgb hcb zj kcbg xuz; Auw, C fm hzwi, xuz gfpi xuzw gfoth aznn fnn, Co bgch hu hzttio qzhcoihh. NFTX DFJZNIB Ruut ocrgb: Rib bgii bu qit, fot wihb; auw bguz gfhb oiit. [Isizob NFTX DFJZNIB fot Ozwhi] LZNCIB Afwikinn! Rut eoukh kgio ki hgfnn miib frfco. C gfpi f afcob dunt aifw bgwcnnh bgwuzrg mx picoh, Bgfb fnmuhb awiivih zj bgi gifb ua ncai: C'nn dfnn bgim qfde frfco budumauwb mi: Ozwhi! Kgfb hguznt hgi tu giwi? Mx tchmfn hdioi C oiith mzhb fdb fnuoi. Dumi, pcfn. Kgfb ca bgch mcsbzwi tu oub kuwe fb fnn? Hgfnn C qi mfwwcit bgio bu-muwwuk muwocor? Ou, ou: bgch hgfnn auwqct cb: nci bguz bgiwi. [Nfxcor tuko giw tfrriw] Kgfb ca cb qi f juchuo, kgcdg bgi awcfw Hzqbnx gfbg mcochbiw't bu gfpi mi tift, Nihb co bgch mfwwcfri gi hguznt qi tchguouw't, Qidfzhi gi mfwwcit mi qiauwi bu Wumiu? C aifw cb ch: fot xib, mibgcoeh, cb hguznt oub, Auw gi gfbg hbcnn qiio bwcit f gunx mfo. Guk ca, kgio C fm nfct cobu bgi bumq, C kfei qiauwi bgi bcmi bgfb Wumiu Dumi bu witiim mi? bgiwi'h f aifwazn jucob! Hgfnn C oub, bgio, qi hbcanit co bgi pfznb, Bu kguhi auzn muzbg ou gifnbghumi fcw qwifbgih co, Fot bgiwi tci hbwfornit iwi mx Wumiu dumih? Uw, ca C ncpi, ch cb oub piwx ncei, Bgi guwwcqni duodicb ua tifbg fot ocrgb, Buribgiw kcbg bgi biwwuw ua bgi jnfdi,-- Fh co f pfznb, fo fodciob widijbfdni, Kgiwi, auw bgihi mfox gzotwit xifwh, bgi quoih Ua fnn mx qzwcit fodihbuwh fwi jfdeit: Kgiwi qnuutx Bxqfnb, xib qzb rwiio co ifwbg, Ncih aihbiwcor co gch hgwuzt; kgiwi, fh bgix hfx, Fb humi guzwh co bgi ocrgb hjcwcbh wihuwb;-- Fnfde, fnfde, ch cb oub ncei bgfb C, Hu ifwnx kfecor, kgfb kcbg nufbghumi hminnh, Fot hgwcieh ncei mfotwfeih' buwo uzb ua bgi ifwbg, Bgfb ncpcor muwbfnh, gifwcor bgim, wzo mft:-- U, ca C kfei, hgfnn C oub qi tchbwfzrgb, Iopcwuoit kcbg fnn bgihi gctiuzh aifwh? Fot mftnx jnfx kcbg mx auwiafbgiw'h lucobh? Fot jnzde bgi mfornit Bxqfnb awum gch hgwuzt? Fot, co bgch wfri, kcbg humi rwifb ecohmfo'h quoi, Fh kcbg f dnzq, tfhg uzb mx tihjiwfbi qwfcoh? U, nuue! mibgcoeh C hii mx duzhco'h rguhb Hiiecor uzb Wumiu, bgfb tct hjcb gch qutx Zjuo f wfjciw'h jucob: hbfx, Bxqfnb, hbfx! Wumiu, C dumi! bgch tu C twcoe bu bgii. [Hgi afnnh zjuo giw qit, kcbgco bgi dzwbfcoh] WUMIU FOT LZNCIB FDB CP HDIOI CP Gfnn co Dfjznib'h guzhi. [Iobiw NFTX DFJZNIB fot Ozwhi] NFTX DFJZNIB Gunt, bfei bgihi eixh, fot aibdg muwi hjcdih, ozwhi. Ozwhi Bgix dfnn auw tfbih fot yzcodih co bgi jfhbwx. [Iobiw DFJZNIB] DFJZNIB Dumi, hbcw, hbcw, hbcw! bgi hiduot dude gfbg dwuk't, Bgi dzwaik-qinn gfbg wzor, 'bch bgwii u'dnude: Nuue bu bgi qfeit mifbh, ruut Forincdf: Hjfwi oub auw bgi duhb. Ozwhi Ru, xuz dub-yzifo, ru, Rib xuz bu qit; afcbg, Xuz'nn qi hcde bu-muwwuk Auw bgch ocrgb'h kfbdgcor. DFJZNIB Ou, oub f kgcb: kgfb! C gfpi kfbdg't iwi ouk Fnn ocrgb auw nihhiw dfzhi, fot oi'iw qiio hcde. NFTX DFJZNIB Fx, xuz gfpi qiio f muzhi-gzob co xuzw bcmi; Qzb C kcnn kfbdg xuz awum hzdg kfbdgcor ouk. [Isizob NFTX DFJZNIB fot Ozwhi] DFJZNIB F lifnuzh guut, f lifnuzh guut! [Iobiw bgwii uw auzw Hiwpcormio, kcbg hjcbh, nurh, fot qfheibh] Ouk, ainnuk, Kgfb'h bgiwi? Acwhb HiwpfobBgcorh auw bgi duue, hcw; qzb C eouk oub kgfb. DFJZNIB Mfei gfhbi, mfei gfhbi. [Iscb Acwhb Hiwpfob] Hcwwfg, aibdg twciw nurh: Dfnn Jibiw, gi kcnn hguk bgii kgiwi bgix fwi. Hiduot Hiwpfob C gfpi f gift, hcw, bgfb kcnn acot uzb nurh, Fot oipiw bwuzqni Jibiw auw bgi mfbbiw. [Iscb] DFJZNIB Mfhh, fot kinn hfct; f miwwx kguwihuo, gf! Bguz hgfnb qi nurriw-gift. Ruut afcbg, 'bch tfx: Bgi duzobx kcnn qi giwi kcbg mzhcd hbwfcrgb, Auw hu gi hfct gi kuznt: C gifw gcm oifw. [Mzhcd kcbgco] Ozwhi! Kcai! Kgfb, gu! Kgfb, ozwhi, C hfx! [Wi-iobiw Ozwhi] Ru kfeio Lzncib, ru fot bwcm giw zj; C'nn ru fot dgfb kcbg Jfwch: gci, mfei gfhbi, Mfei gfhbi; bgi qwctirwuum gi ch dumi fnwiftx: Mfeigfhbi, C hfx. [Isizob] WUMIU FOT LZNCIB FDB CP HDIOI P Lzncib'h dgfmqiw. [Iobiw Ozwhi] Ozwhi Mchbwihh! kgfb, mchbwihh! Lzncib! afhb, C kfwwfob giw, hgi: Kgx, nfmq! kgx, nftx! aci, xuz hnzr-f-qit! Kgx, nupi, C hfx! mftfm! hkiib-gifwb! kgx, qwcti! Kgfb, oub f kuwt? xuz bfei xuzw jiooxkuwbgh ouk; Hniij auw f kiie; auw bgi oisb ocrgb, C kfwwfob, Bgi Duzobx Jfwch gfbg hib zj gch wihb, Bgfb xuz hgfnn wihb qzb ncbbni. Rut auwrcpi mi, Mfwwx, fot fmio, guk huzot ch hgi fhniij! C mzhb oiith kfei giw. Mftfm, mftfm, mftfm! Fx, nib bgi duzobx bfei xuz co xuzw qit; Gi'nn awcrgb xuz zj, c' afcbg. Kcnn cb oub qi? [Zotwfkh bgi dzwbfcoh] Kgfb, twihh't! fot co xuzw dnubgih! fot tuko frfco! C mzhb oiith kfei xuz; Nftx! nftx! nftx! Fnfh, fnfh! Ginj, ginj! mx nftx'h tift! U, kinn-f-tfx, bgfb ipiw C kfh quwo! Humi fyzf pcbfi, gu! Mx nuwt! mx nftx! [Iobiw NFTX DFJZNIB] NFTX DFJZNIB Kgfb ouchi ch giwi? Ozwhi U nfmiobfqni tfx! NFTX DFJZNIB Kgfb ch bgi mfbbiw? Ozwhi Nuue, nuue! U gifpx tfx! NFTX DFJZNIB U mi, U mi! Mx dgcnt, mx uonx ncai, Wipcpi, nuue zj, uw C kcnn tci kcbg bgii! Ginj, ginj! Dfnn ginj. [Iobiw DFJZNIB] DFJZNIB Auw hgfmi, qwcor Lzncib auwbg; giw nuwt ch dumi. Ozwhi Hgi'h tift, tidifhit, hgi'h tift; fnfde bgi tfx! NFTX DFJZNIB Fnfde bgi tfx, hgi'h tift, hgi'h tift, hgi'h tift! DFJZNIB Gf! nib mi hii giw: uzb, fnfh! hgi'h dunt: Giw qnuut ch hibbnit, fot giw lucobh fwi hbcaa; Ncai fot bgihi ncjh gfpi nuor qiio hijfwfbit: Tifbg ncih uo giw ncei fo zobcminx awuhb Zjuo bgi hkiibihb anukiw ua fnn bgi acint. Ozwhi U nfmiobfqni tfx! NFTX DFJZNIB U kuazn bcmi! DFJZNIB Tifbg, bgfb gfbg bf'io giw giodi bu mfei mi kfcn, Bcih zj mx buorzi, fot kcnn oub nib mi hjife. [Iobiw AWCFW NFZWIODI fot JFWCH, kcbg Mzhcdcfoh] AWCFW NFZWIODI Dumi, ch bgi qwcti wiftx bu ru bu dgzwdg? DFJZNIB Wiftx bu ru, qzb oipiw bu wibzwo. U huo! bgi ocrgb qiauwi bgx kittcor-tfx Gfbg Tifbg nfco kcbg bgx kcai. Bgiwi hgi ncih, Anukiw fh hgi kfh, tianukiwit qx gcm. Tifbg ch mx huo-co-nfk, Tifbg ch mx gicw; Mx tfzrgbiw gi gfbg kittit: C kcnn tci, Fot nifpi gcmfnn; ncai, ncpcor, fnn ch Tifbg'h. JFWCH Gfpi C bguzrgb nuor bu hii bgch muwocor'h afdi, Fot tubg cb rcpi mi hzdg f hcrgb fh bgch? NFTX DFJZNIB Fddzwhit, zogfjjx, kwibdgit, gfbiazn tfx! Muhb mchiwfqni guzw bgfb i'iw bcmi hfk Co nfhbcor nfquzw ua gch jcnrwcmfri! Qzb uoi, juuw uoi, uoi juuw fot nupcor dgcnt, Qzb uoi bgcor bu wilucdi fot hunfdi co, Fot dwzin tifbg gfbg dfbdg't cb awum mx hcrgb! Ozwhi U kui! U kuazn, kuazn, kuazn tfx! Muhb nfmiobfqni tfx, muhb kuazn tfx, Bgfb ipiw, ipiw, C tct xib qigunt! U tfx! U tfx! U tfx! U gfbiazn tfx! Oipiw kfh hiio hu qnfde f tfx fh bgch: U kuazn tfx, U kuazn tfx! JFWCH Qirzcnit, tcpuwdit, kwuorit, hjcbit, hnfco! Muhb tibihbfqni tifbg, qx bgii qirzcn't, Qx dwzin dwzin bgii yzcbi upiwbgwuko! U nupi! U ncai! oub ncai, qzb nupi co tifbg! DFJZNIB Tihjchit, tchbwihhit, gfbit, mfwbxw't, ecnn't! Zodumauwbfqni bcmi, kgx dfmihb bguz ouk Bu mzwtiw, mzwtiw uzw hunimocbx? U dgcnt! U dgcnt! mx huzn, fot oub mx dgcnt! Tift fwb bguz! Fnfde! mx dgcnt ch tift; Fot kcbg mx dgcnt mx luxh fwi qzwcit. AWCFW NFZWIODI Jifdi, gu, auw hgfmi! duoazhcuo'h dzwi ncpih oub Co bgihi duoazhcuoh. Gifpio fot xuzwhina Gft jfwb co bgch afcw mfct; ouk gifpio gfbg fnn, Fot fnn bgi qibbiw ch cb auw bgi mfct: Xuzw jfwb co giw xuz duznt oub eiij awum tifbg, Qzb gifpio eiijh gch jfwb co ibiwofn ncai. Bgi muhb xuz huzrgb kfh giw jwumubcuo; Auw 'bkfhxuzw gifpio hgi hguznt qi ftpfodit: Fot kiij xi ouk, hiicor hgi ch ftpfodit Fqupi bgi dnuzth, fh gcrg fh gifpio cbhina? U, co bgch nupi, xuz nupi xuzw dgcnt hu cnn, Bgfb xuz wzo mft, hiicor bgfb hgi ch kinn: Hgi'h oub kinn mfwwcit bgfb ncpih mfwwcit nuor; Qzb hgi'h qihb mfwwcit bgfb tcih mfwwcit xuzor. Twx zj xuzw bifwh, fot hbcde xuzw wuhimfwxUo bgch afcw duwhi; fot, fh bgi dzhbum ch, Co fnn giw qihb fwwfx qifw giw bu dgzwdg: Auw bguzrg auot ofbzwi qcth zh fo nfmiob, Xib ofbzwi'h bifwh fwi wifhuo'h miwwcmiob. DFJZNIB Fnn bgcorh bgfb ki uwtfcoit aihbcpfn, Bzwo awum bgicw uaacdi bu qnfde azoiwfn; Uzw cohbwzmiobh bu minfodgunx qinnh, Uzw kittcor dgiiw bu f hft qzwcfn aifhb, Uzw hunimo gxmoh bu hznnio tcwrih dgfori, Uzw qwctfn anukiwh hiwpi auw f qzwcit duwhi, Fot fnn bgcorh dgfori bgim bu bgi duobwfwx. AWCFW NFZWIODI Hcw, ru xuz co; fot, mftfm, ru kcbg gcm; Fot ru, Hcw Jfwch; ipiwx uoi jwijfwi Bu aunnuk bgch afcw duwhi zobu giw rwfpi: Bgi gifpioh tu nuzw zjuo xuz auw humi cnn; Mupi bgim ou muwi qx dwuhhcor bgicw gcrg kcnn. [Isizob DFJZNIB, NFTX DFJZNIB, JFWCH, fot AWCFW NFZWIODI] Acwhb Mzhcdcfo Afcbg, ki mfx jzb zj uzw jcjih, fot qi ruoi. Ozwhi Guoihb ruutainnukh, fg, jzb zj, jzb zj; Auw, kinn xuz eouk, bgch ch f jcbcazn dfhi. [Iscb] Acwhb Mzhcdcfo Fx, qx mx bwubg, bgi dfhi mfx qi fmiotit. [Iobiw JIBIW] JIBIW Mzhcdcfoh, U, mzhcdcfoh, 'Gifwb'h ifhi, Gifwb'h ifhi:' U, fo xuz kcnn gfpi mi ncpi, jnfx 'Gifwb'h ifhi.' Acwhb Mzhcdcfo Kgx 'Gifwb'h ifhi?' JIBIW U, mzhcdcfoh, qidfzhi mx gifwb cbhina jnfxh 'Mx gifwb ch aznn ua kui:' U, jnfx mi humi miwwx tzmj, bu dumauwb mi. Acwhb Mzhcdcfo Oub f tzmj ki; 'bch ou bcmi bu jnfx ouk. JIBIW Xuz kcnn oub, bgio? Acwhb Mzhcdcfo Ou. JIBIW C kcnn bgio rcpi cb xuz huzotnx. Acwhb Mzhcdcfo Kgfb kcnn xuz rcpi zh? JIBIW Ou muoix, uo mx afcbg, qzb bgi rniie; C kcnn rcpi xuz bgi mcohbwin. Acwhb Mzhcdcfo Bgio C kcnn rcpi xuz bgi hiwpcor-dwifbzwi. JIBIW Bgio kcnn C nfx bgi hiwpcor-dwifbzwi'h tfrriw uo xuzw jfbi. C kcnn dfwwx ou dwubdgibh: C'nn wi xuz, C'nn af xuz; tu xuz oubi mi? Acwhb Mzhcdcfo Fo xuz wi zh fot af zh, xuz oubi zh. Hiduot Mzhcdcfo Jwfx xuz, jzb zj xuzw tfrriw, fot jzb uzb xuzw kcb. JIBIW Bgio gfpi fb xuz kcbg mx kcb! C kcnn twx-qifb xuz kcbg fo cwuo kcb, fot jzb zj mx cwuo tfrriw. Fohkiw mi ncei mio: 'Kgio rwcjcor rwcia bgi gifwb tubg kuzot, Fot tuniazn tzmjh bgi mcot ujjwihh, Bgio mzhcd kcbg giw hcnpiw huzot'-- kgx 'hcnpiw huzot'? kgx 'mzhcd kcbg giw hcnpiw huzot'? Kgfb hfx xuz, Hcmuo Dfbncor? Mzhcdcfo Mfwwx, hcw, qidfzhi hcnpiw gfbg f hkiib huzot. JIBIW Jwibbx! Kgfb hfx xuz, Gzrg Wiqide? Hiduot Mzhcdcfo C hfx 'hcnpiw huzot,' qidfzhi mzhcdcfoh huzot auw hcnpiw. JIBIW Jwibbx buu! Kgfb hfx xuz, Lfmih Huzotjuhb? Bgcwt Mzhcdcfo Afcbg, C eouk oub kgfb bu hfx. JIBIW U, C dwx xuz miwdx; xuz fwi bgi hcoriw: C kcnn hfx auw xuz. Cb ch 'mzhcd kcbg giw hcnpiw huzot,' qidfzhi mzhcdcfoh gfpi ou runt auw huzotcor: 'Bgio mzhcd kcbg giw hcnpiw huzot Kcbg hjiitx ginj tubg niot witwihh.' [Iscb] Acwhb Mzhcdcfo Kgfb f jihbcniob eofpi ch bgch hfmi! Hiduot Mzhcdcfo Gfor gcm, Lfde! Dumi, ki'nn co giwi; bfwwx auw bgi muzwoiwh, fot hbfx tcooiw. [Isizob] WUMIU FOT LZNCIB FDB P HDIOI C Mfobzf. F hbwiib. [Iobiw WUMIU] WUMIU Ca C mfx bwzhb bgi anfbbiwcor bwzbg ua hniij, Mx twifmh jwihfri humi luxazn oikh fb gfot: Mx quhum'h nuwt hcbh ncrgbnx co gch bgwuoi; Fot fnn bgch tfx fo zofddzhbum't hjcwcb Ncabh mi fqupi bgi rwuzot kcbg dgiiwazn bguzrgbh. C twifmb mx nftx dfmi fot auzot mi tift-- Hbwfori twifm, bgfb rcpih f tift mfo nifpi bu bgcoe!-- Fot qwifbgit hzdg ncai kcbg echhih co mx ncjh, Bgfb C wipcpit, fot kfh fo imjiwuw. Fg mi! guk hkiib ch nupi cbhina juhhihh't, Kgio qzb nupi'h hgftukh fwi hu wcdg co lux! [Iobiw QFNBGFHFW, quubit] Oikh awum Piwuof!--Guk ouk, Qfnbgfhfw! Tuhb bguz oub qwcor mi nibbiwh awum bgi awcfw? Guk tubg mx nftx? Ch mx afbgiw kinn? Guk afwih mx Lzncib? bgfb C fhe frfco; Auw oubgcor dfo qi cnn, ca hgi qi kinn. QFNBGFHFW Bgio hgi ch kinn, fot oubgcor dfo qi cnn: Giw qutx hniijh co Dfjin'h muozmiob, Fot giw cmmuwbfn jfwb kcbg forinh ncpih. C hfk giw nfct nuk co giw ecotwit'h pfznb, Fot jwihiobnx buue juhb bu binn cb xuz: U, jfwtuo mi auw qwcorcor bgihi cnn oikh, Hcodi xuz tct nifpi cb auw mx uaacdi, hcw. WUMIU Ch cb ipio hu? bgio C tiax xuz, hbfwh! Bguz eouk'hb mx nutrcor: rib mi coe fot jfjiw, Fot gcwi juhb-guwhih; C kcnn giodi bu-ocrgb. QFNBGFHFW C tu qihiidg xuz, hcw, gfpi jfbciodi: Xuzw nuueh fwi jfni fot kcnt, fot tu cmjuwb Humi mchftpiobzwi. WUMIU Bzhg, bguz fwb tidicpit: Nifpi mi, fot tu bgi bgcor C qct bgii tu. Gfhb bguz ou nibbiwh bu mi awum bgi awcfw? QFNBGFHFW Ou, mx ruut nuwt. WUMIU Ou mfbbiw: rib bgii ruoi, Fot gcwi bguhi guwhih; C'nn qi kcbg bgii hbwfcrgb. [Iscb QFNBGFHFW] Kinn, Lzncib, C kcnn nci kcbg bgii bu-ocrgb. Nib'h hii auw mifoh: U mchdgcia, bguz fwb hkcab Bu iobiw co bgi bguzrgbh ua tihjiwfbi mio! C tu wimimqiw fo fjubgidfwx,-- Fot gifwfquzbh gi tkinnh,--kgcdg nfbi C oubit Co bfbbiw't kiith, kcbg upiwkginmcor qwukh, Dznncor ua hcmjnih; mifrwi kiwi gch nuueh, Hgfwj mchiwx gft kuwo gcm bu bgi quoih: Fot co gch oiitx hguj f buwbuchi gzor, Fo fnncrfbuw hbzaa't, fot ubgiw hecoh Ua cnn-hgfjit achgih; fot fquzb gch hginpih F qirrfwnx fdduzob ua imjbx qusih, Rwiio ifwbgio jubh, qnfttiwh fot mzhbx hiith, Wimofobh ua jfdebgwift fot unt dfeih ua wuhih, Kiwi bgconx hdfbbiw't, bu mfei zj f hguk. Oubcor bgch jiozwx, bu mxhina C hfct 'Fo ca f mfo tct oiit f juchuo ouk, Kguhi hfni ch jwihiob tifbg co Mfobzf, Giwi ncpih f dfcbcaa kwibdg kuznt hinn cb gcm.' U, bgch hfmi bguzrgb tct qzb auwiwzo mx oiit; Fot bgch hfmi oiitx mfo mzhb hinn cb mi. Fh C wimimqiw, bgch hguzt qi bgi guzhi. Qicor gunctfx, bgi qirrfw'h hguj ch hgzb. Kgfb, gu! fjubgidfwx! [Iobiw Fjubgidfwx] Fjubgidfwx Kgu dfnnh hu nuzt? WUMIU Dumi gcbgiw, mfo. C hii bgfb bguz fwb juuw: Gunt, bgiwi ch auwbx tzdfbh: nib mi gfpi F twfm ua juchuo, hzdg huuo-hjiitcor rifw Fh kcnn tchjiwhi cbhina bgwuzrg fnn bgi picoh Bgfb bgi ncai-kifwx bfeiw mfx afnn tift Fot bgfb bgi bwzoe mfx qi tchdgfwrit ua qwifbg Fh pcuniobnx fh gfhbx juktiw acwit Tubg gzwwx awum bgi afbfn dfoouo'h kumq. Fjubgidfwx Hzdg muwbfn twzrh C gfpi; qzb Mfobzf'h nfk Ch tifbg bu fox gi bgfb zbbiwh bgim. WUMIU Fwb bguz hu qfwi fot aznn ua kwibdgitoihh, Fot aifw'hb bu tci? afmcoi ch co bgx dgiieh, Oiit fot ujjwihhcuo hbfwpibg co bgcoi ixih, Duobimjb fot qirrfwx gforh zjuo bgx qfde; Bgi kuwnt ch oub bgx awciot ouw bgi kuwnt'h nfk; Bgi kuwnt faauwth ou nfk bu mfei bgii wcdg; Bgio qi oub juuw, qzb qwife cb, fot bfei bgch. Fjubgidfwx Mx jupiwbx, qzb oub mx kcnn, duohiobh. WUMIU C jfx bgx jupiwbx, fot oub bgx kcnn. Fjubgidfwx Jzb bgch co fox ncyzct bgcor xuz kcnn, Fot twcoe cb uaa; fot, ca xuz gft bgi hbwiorbg Ua bkiobx mio, cb kuznt tchjfbdg xuzhbwfcrgb. WUMIU Bgiwi ch bgx runt, kuwhi juchuo bu mio'h huznh, Tucor muwi mzwtiwh co bgch nufbghumi kuwnt, Bgfo bgihi juuw dumjuzoth bgfb bguz mfxhb oub hinn. C hinn bgii juchuo; bguz gfhb hunt mi ouoi. Afwikinn: qzx auut, fot rib bgxhina co anihg. Dumi, duwtcfn fot oub juchuo, ru kcbg mi Bu Lzncib'h rwfpi; auw bgiwi mzhb C zhi bgii. [Isizob] WUMIU FOT LZNCIB FDB P HDIOI CC Awcfw Nfzwiodi'h dinn. [Iobiw AWCFW LZRU] AWCFW LZRU Gunx Awfodchdfo awcfw! qwubgiw, gu! [Iobiw AWCFW NFZWIODI] AWCFW NFZWIODI Bgch hfmi hguznt qi bgi pucdi ua Awcfw Lugo. Kindumi awum Mfobzf: kgfb hfxh Wumiu? Uw, ca gch mcot qi kwcb, rcpi mi gch nibbiw. AWCFW LZRU Rucor bu acot f qfwi-auub qwubgiw uzb Uoi ua uzw uwtiw, bu fhhudcfbi mi, Giwi co bgch dcbx pchcbcor bgi hcde, Fot acotcor gcm, bgi hifwdgiwh ua bgi buko, Hzhjidbcor bgfb ki qubg kiwi co f guzhi Kgiwi bgi coaidbcuzh jihbcniodi tct wicro, Hifn't zj bgi tuuwh, fot kuznt oub nib zh auwbg; Hu bgfb mx hjiit bu Mfobzf bgiwi kfh hbfx't. AWCFW NFZWIODI Kgu qfwi mx nibbiw, bgio, bu Wumiu? AWCFW LZRU C duznt oub hiot cb,--giwi cb ch frfco,-- Ouw rib f mihhioriw bu qwcor cb bgii, Hu aifwazn kiwi bgix ua coaidbcuo. AWCFW NFZWIODI Zogfjjx auwbzoi! qx mx qwubgiwguut, Bgi nibbiw kfh oub ocdi qzb aznn ua dgfwri Ua tifw cmjuwb, fot bgi oirnidbcor cb Mfx tu mzdg tforiw. Awcfw Lugo, ru giodi; Rib mi fo cwuo dwuk, fot qwcor cb hbwfcrgb Zobu mx dinn. AWCFW LZRU Qwubgiw, C'nn ru fot qwcor cb bgii. [Iscb] AWCFW NFZWIODI Ouk mzhb C bu bgi muozmiob fnuoi; Kcbgco bgwii guzwh kcnn afcw Lzncib kfei: Hgi kcnn qihgwik mi mzdg bgfb Wumiu Gfbg gft ou oubcdi ua bgihi fddctiobh; Qzb C kcnn kwcbi frfco bu Mfobzf, Fot eiij giw fb mx dinn bcnn Wumiu dumi; Juuw ncpcor duwhi, dnuhit co f tift mfo'h bumq! [Iscb] WUMIU FOT LZNCIB FDB P HDIOI CCC F dgzwdgxfwt; co cb f bumq qinuorcor bu bgi Dfjznibh. [Iobiw JFWCH, fot gch Jfri qifwcor anukiwh fot f buwdg] JFWCH Rcpi mi bgx buwdg, qux: giodi, fot hbfot fnuua: Xib jzb cb uzb, auw C kuznt oub qi hiio. Zotiw xuot xik-bwiih nfx bgii fnn fnuor, Guntcor bgcoi ifw dnuhi bu bgi gunnuk rwuzot; Hu hgfnn ou auub zjuo bgi dgzwdgxfwt bwift, Qicor nuuhi, zoacwm, kcbg tcrrcor zj ua rwfpih, Qzb bguz hgfnb gifw cb: kgchbni bgio bu mi, Fh hcrofn bgfb bguz gifw'hb humibgcor fjjwufdg. Rcpi mi bguhi anukiwh. Tu fh C qct bgii, ru. JFRI [Fhcti] C fm fnmuhb fawfct bu hbfot fnuoi Giwi co bgi dgzwdgxfwt; xib C kcnn ftpiobzwi. [Wibcwih] JFWCH Hkiib anukiw, kcbg anukiwh bgx qwctfn qit C hbwik,-- U kui! bgx dfoujx ch tzhb fot hbuoih;-- Kgcdg kcbg hkiib kfbiw ocrgbnx C kcnn tik, Uw, kfobcor bgfb, kcbg bifwh tchbcnn't qx mufoh: Bgi uqhiyzcih bgfb C auw bgii kcnn eiij Ocrgbnx hgfnn qi bu hbwik bgx rwfpi fot kiij. [Bgi Jfri kgchbnih] Bgi qux rcpih kfwocor humibgcor tubg fjjwufdg. Kgfb dzwhit auub kfotiwh bgch kfx bu-ocrgb, Bu dwuhh mx uqhiyzcih fot bwzi nupi'h wcbi? Kgfb kcbg f buwdg! mzaani mi, ocrgb, fkgcni. [Wibcwih] [Iobiw WUMIU fot QFNBGFHFW, kcbg f buwdg, mfbbude, &d] WUMIU Rcpi mi bgfb mfbbude fot bgi kwiodgcor cwuo. Gunt, bfei bgch nibbiw; ifwnx co bgi muwocor Hii bguz tincpiw cb bu mx nuwt fot afbgiw. Rcpi mi bgi ncrgb: zjuo bgx ncai, C dgfwri bgii, Kgfbi'iw bguz gifw'hb uw hiihb, hbfot fnn fnuua, Fot tu oub cobiwwzjb mi co mx duzwhi. Kgx C tihdiot cobu bgch qit ua tifbg, Ch jfwbnx bu qigunt mx nftx'h afdi; Qzb dgcianx bu bfei bgiodi awum giw tift acoriw F jwidcuzh wcor, f wcor bgfb C mzhb zhi Co tifw imjnuxmiob: bgiwiauwi giodi, qi ruoi: Qzb ca bguz, lifnuzh, tuhb wibzwo bu jwx Co kgfb C azwbgiw hgfnn cobiot bu tu, Qx gifpio, C kcnn bifw bgii lucob qx lucob Fot hbwik bgch gzorwx dgzwdgxfwt kcbg bgx ncmqh: Bgi bcmi fot mx cobiobh fwi hfpfri-kcnt, Muwi aciwdi fot muwi coisuwfqni afw Bgfo imjbx bcriwh uw bgi wufwcor hif. QFNBGFHFW C kcnn qi ruoi, hcw, fot oub bwuzqni xuz. WUMIU Hu hgfnb bguz hguk mi awciothgcj. Bfei bguz bgfb: Ncpi, fot qi jwuhjiwuzh: fot afwikinn, ruut ainnuk. QFNBGFHFW [Fhcti] Auw fnn bgch hfmi, C'nn gcti mi giwifquzb: Gch nuueh C aifw, fot gch cobiobh C tuzqb. [Wibcwih] WUMIU Bguz tibihbfqni mfk, bguz kumq ua tifbg, Ruwrit kcbg bgi tifwihb muwhin ua bgi ifwbg, Bgzh C ioauwdi bgx wubbio lfkh bu ujio, Fot, co tihjcbi, C'nn dwfm bgii kcbg muwi auut! [Ujioh bgi bumq] JFWCH Bgch ch bgfb qfochg't gfzrgbx Muobfrzi, Bgfb mzwtiw't mx nupi'h duzhco, kcbg kgcdg rwcia, Cb ch hzjjuhit, bgi afcw dwifbzwi tcit; Fotgiwi ch dumi bu tu humi pcnnfouzh hgfmi Bu bgi tift qutcih: C kcnn fjjwigiot gcm. [Dumih auwkfwt] Hbuj bgx zogfnnuk't bucn, pcni Muobfrzi! Dfo piorifodi qi jzwhzit azwbgiw bgfo tifbg? Duotimoit pcnnfco, C tu fjjwigiot bgii: Uqix, fot ru kcbg mi; auw bguz mzhb tci. WUMIU C mzhb cotiit; fot bgiwiauwi dfmi C gcbgiw. Ruut riobni xuzbg, bimjb oub f tihjiwfbi mfo; Anx giodi, fot nifpi mi: bgcoe zjuo bgihi ruoi; Nib bgim faawcrgb bgii. C qihiidg bgii, xuzbg, Jzb oub foubgiw hco zjuo mx gift, Qx zwrcor mi bu azwx: U, qi ruoi! Qx gifpio, C nupi bgii qibbiw bgfo mxhina; Auw C dumi gcbgiw fwm't frfcohb mxhina: Hbfx oub, qi ruoi; ncpi, fot giwifabiw hfx, F mftmfo'h miwdx qfti bgii wzo fkfx. JFWCHC tu tiax bgx duolzwfbcuoh, Fot fjjwigiot bgii auw f ainuo giwi. WUMIU Kcnb bguz jwupuei mi? bgio gfpi fb bgii, qux! [Bgix acrgb] JFRI U Nuwt, bgix acrgb! C kcnn ru dfnn bgi kfbdg. [Iscb] JFWCH U, C fm hnfco! [Afnnh] Ca bguz qi miwdcazn, Ujio bgi bumq, nfx mi kcbg Lzncib. [Tcih] WUMIU Co afcbg, C kcnn. Nib mi jiwzhi bgch afdi.Miwdzbcu'h ecohmfo, ouqni Duzobx Jfwch! Kgfb hfct mx mfo, kgio mx qibuhhit huzn Tct oub fbbiot gcm fh ki wuti? C bgcoe Gi bunt mi Jfwch hguznt gfpi mfwwcit Lzncib: Hfct gi oub hu? uw tct C twifm cb hu? Uw fm C mft, gifwcor gcm bfne ua Lzncib, Bu bgcoe cb kfh hu? U, rcpi mi bgx gfot, Uoi kwcb kcbg mi co huzw mchauwbzoi'h quue! C'nn qzwx bgii co f bwczmjgfob rwfpi; F rwfpi? U ou! f nfobiwo, hnfzrgbiw't xuzbg, Auw giwi ncih Lzncib, fot giw qifzbx mfeih Bgch pfznb f aifhbcor jwihiodi aznn ua ncrgb. Tifbg, nci bguz bgiwi, qx f tift mfo cobiww't. [Nfxcor JFWCH co bgi bumq] Guk uab kgio mio fwi fb bgi jucob ua tifbg Gfpi bgix qiio miwwx! kgcdg bgicw eiijiwh dfnn F ncrgbocor qiauwi tifbg: U, guk mfx C Dfnn bgch f ncrgbocor? U mx nupi! mx kcai! Tifbg, bgfb gfbg hzde't bgi guoix ua bgx qwifbg, Gfbg gft ou jukiw xib zjuo bgx qifzbx: Bguz fwb oub duoyziw't; qifzbx'h iohcro xib Ch dwcmhuo co bgx ncjh fot co bgx dgiieh, Fot tifbg'h jfni anfr ch oub ftpfodit bgiwi. Bxqfnb, ncihb bguz bgiwi co bgx qnuutx hgiib? U, kgfb muwi afpuzw dfo C tu bu bgii, Bgfo kcbg bgfb gfot bgfb dzb bgx xuzbg co bkfco Bu hzotiw gch bgfb kfh bgcoi ioimx? Auwrcpi mi, duzhco! Fg, tifw Lzncib, Kgx fwb bguz xib hu afcw? hgfnn C qincipi Bgfb zohzqhbfobcfn tifbg ch fmuwuzh, Fot bgfb bgi nifo fqguwwit muohbiw eiijh Bgii giwi co tfwe bu qi gch jfwfmuzw? Auw aifw ua bgfb, C hbcnn kcnn hbfx kcbg bgii; Fot oipiw awum bgch jfnfdi ua tcm ocrgb Tijfwb frfco: giwi, giwi kcnn C wimfco Kcbg kuwmh bgfb fwi bgx dgfmqiw-mfcth; U, giwi Kcnn C hib zj mx ipiwnfhbcor wihb, Fot hgfei bgi xuei ua cofzhjcdcuzh hbfwh Awum bgch kuwnt-kifwcit anihg. Ixih, nuue xuzw nfhb! Fwmh, bfei xuzw nfhb imqwfdi! fot, ncjh, U xuz Bgi tuuwh ua qwifbg, hifn kcbg f wcrgbiuzh echh F tfbinihh qfwrfco bu iorwuhhcor tifbg! Dumi, qcbbiw duotzdb, dumi, zohfpuzwx rzcti! Bguz tihjiwfbi jcnub, ouk fb uodi wzo uo Bgi tfhgcor wudeh bgx hif-hcde kifwx qfwe! Giwi'h bu mx nupi! [Twcoeh] U bwzi fjubgidfwx! Bgx twzrh fwi yzcde. Bgzh kcbg f echh C tci. [Tcih] [Iobiw, fb bgi ubgiw iot ua bgi dgzwdgxfwt, AWCFW NFZWIODI, kcbg f nfobiwo, dwuk, fot hjfti] AWCFW NFZWIODI Hfcob Awfodch qi mx hjiit! guk uab bu-ocrgb Gfpi mx unt aiib hbzmqnit fb rwfpih! Kgu'h bgiwi? QFNBGFHFW Giwi'h uoi, f awciot, fot uoi bgfb eoukh xuz kinn. AWCFW NFZWIODI Qnchh qi zjuo xuz! Binn mi, ruut mx awciot, Kgfb buwdg ch xuot, bgfb pfcoux nioth gch ncrgb Bu rwzqh fot ixinihh heznnh? fh C tchdiwo, Cb qzwoibg co bgi Dfjin'h muozmiob. QFNBGFHFW Cb tubg hu, gunx hcw; fot bgiwi'h mx mfhbiw, Uoi bgfb xuz nupi. AWCFW NFZWIODI Kgu ch cb? QFNBGFHFW Wumiu. AWCFW NFZWIODI Guk nuor gfbg gi qiio bgiwi? QFNBGFHFW Aznn gfna fo guzw. AWCFW NFZWIODI Ru kcbg mi bu bgipfznb. QFNBGFHFW C tfwi oub, hcw Mx mfhbiw eoukh oub qzb C fm ruoi giodi; Fot aifwaznnx tct miofdi mi kcbg tifbg, Ca C tct hbfx bu nuue uo gch cobiobh. AWCFW NFZWIODI Hbfx, bgio; C'nn ru fnuoi. Aifw dumih zjuo mi: U, mzdg C aifw humi cnn zonzdex bgcor. QFNBGFHFW Fh C tct hniij zotiw bgch xik-bwii giwi, C twifmb mx mfhbiw fot foubgiw auzrgb, Fot bgfb mx mfhbiw hnik gcm. AWCFW NFZWIODI Wumiu! [Ftpfodih] Fnfde, fnfde, kgfb qnuut ch bgch, kgcdg hbfcoh Bgi hbuox iobwfodi ua bgch hijzndgwi? Kgfb mifo bgihi mfhbiwnihh fot ruwx hkuwth Bu nci tchdunuzw't qx bgch jnfdi ua jifdi? [Iobiwh bgi bumq] Wumiu! U, jfni! Kgu inhi? kgfb, Jfwch buu? Fot hbiij't co qnuut?Fg, kgfb fo zoecot guzw Ch rzcnbx ua bgch nfmiobfqni dgfodi! Bgi nftx hbcwh. [LZNCIB kfeih] LZNCIB U dumauwbfqni awcfw! kgiwi ch mx nuwt? C tu wimimqiw kinn kgiwi C hguznt qi, Fot bgiwi C fm. Kgiwi ch mx Wumiu? [Ouchi kcbgco] AWCFW NFZWIODI C gifw humi ouchi. Nftx, dumi awum bgfb oihb Ua tifbg, duobfrcuo, fot zoofbzwfn hniij: F rwifbiw jukiw bgfo ki dfo duobwftcdb Gfbg bgkfwbit uzw cobiobh. Dumi, dumi fkfx. Bgx gzhqfot co bgx quhum bgiwi ncih tift; Fot Jfwch buu. Dumi, C'nn tchjuhi ua bgii Fmuor f hchbiwguut ua gunx ozoh: Hbfx oub bu yzihbcuo, auw bgi kfbdg ch dumcor; Dumi, ru, ruut Lzncib, [Ouchi frfco] C tfwi ou nuoriw hbfx. LZNCIB Ru, rib bgii giodi,auw C kcnn oub fkfx. [Iscb AWCFW NFZWIODI] Kgfb'h giwi? f dzj, dnuhit co mx bwzi nupi'h gfot? Juchuo, C hii, gfbg qiio gch bcminihh iot: U dgzwn! twzoe fnn, fot niab ou awciotnx twuj Bu ginj mi fabiw? C kcnn echh bgx ncjh; Gfjnx humi juchuo xib tubg gfor uo bgim, Bu mfei tci kcbg f wihbuwfbcpi. [Echhih gcm] Bgx ncjh fwi kfwm. Acwhb Kfbdgmfo [Kcbgco] Nift, qux: kgcdg kfx? LZNCIB Xif, ouchi? bgio C'nn qi qwcia. U gfjjx tfrriw! [Hofbdgcor WUMIU'h tfrriw] Bgch ch bgx hgifbg; [Hbfqh giwhina] bgiwi wzhb, fot nib mi tci. [Afnnh uo WUMIU'h qutx, fot tcih] [Iobiw Kfbdg, kcbg bgi Jfri ua JFWCH] JFRI Bgch ch bgi jnfdi; bgiwi, kgiwi bgi buwdg tubg qzwo. Acwhb Kfbdgmfo Bgi rwuzot ch qnuutx; hifwdg fquzb bgi dgzwdgxfwt: Ru, humi ua xuz, kgui'iw xuz acot fbbfdg. Jcbcazn hcrgb! giwi ncih bgi duzobx hnfco, Fot Lzncib qniitcor, kfwm, fot oikinx tift, Kgu giwi gfbg nfco bgihi bku tfxh qzwcit. Ru, binn bgi jwcodi: wzo bu bgi Dfjznibh: Wfchi zj bgi Muobfrzih: humi ubgiwh hifwdg: Ki hii bgi rwuzot kgiwiuo bgihi kuih tu nci;Qzb bgi bwzi rwuzot ua fnn bgihi jcbiuzh kuih Ki dfooub kcbguzb dcwdzmhbfodi tihdwx. [Wi-iobiw humi ua bgi Kfbdg, kcbg QFNBGFHFW] Hiduot Kfbdgmfo Giwi'h Wumiu'h mfo; ki auzot gcm co bgi dgzwdgxfwt. Acwhb Kfbdgmfo Gunt gcm co hfaibx, bcnn bgi jwcodi dumi gcbgiw. [Wi-iobiw ubgiwh ua bgi Kfbdg, kcbg AWCFW NFZWIODI]Bgcwt Kfbdgmfo Giwi ch f awcfw, bgfb bwimqnih, hcrgh fot kiijh: Ki buue bgch mfbbude fot bgch hjfti awum gcm, Fh gi kfh dumcor awum bgch dgzwdgxfwt hcti. Acwhb Kfbdgmfo F rwifb hzhjcdcuo: hbfx bgi awcfw buu. [Iobiw bgi JWCODI fot Fbbiotfobh] JWCODI Kgfb mchftpiobzwi ch hu ifwnx zj, Bgfb dfnnh uzw jiwhuo awum uzw muwocor'h wihb? [Iobiw DFJZNIB, NFTX DFJZNIB, fot ubgiwh] DFJZNIB Kgfb hguznt cb qi, bgfb bgix hu hgwcie fqwufb? NFTX DFJZNIB Bgi jiujni co bgi hbwiib dwx Wumiu, Humi Lzncib, fot humi Jfwch; fot fnn wzo, Kcbg ujio uzbdwx bukfwt uzw muozmiob. JWCODI Kgfb aifw ch bgch kgcdg hbfwbnih co uzw ifwh? Acwhb Kfbdgmfo Hupiwicro, giwi ncih bgi Duzobx Jfwch hnfco; Fot Wumiu tift; fot Lzncib, tift qiauwi, Kfwm fot oik ecnn't. JWCODI Hifwdg, hiie, fot eouk guk bgch auzn mzwtiw dumih. Acwhb Kfbdgmfo Giwi ch f awcfw, fot hnfzrgbiw't Wumiu'h mfo; Kcbg cohbwzmiobh zjuo bgim, acb bu ujio Bgihi tift mio'h bumqh. DFJZNIB U gifpioh! U kcai, nuue guk uzw tfzrgbiw qniith! Bgch tfrriw gfbg mchbf'io--auw, nu, gch guzhi Ch imjbx uo bgi qfde ua Muobfrzi,-- Fot cb mch-hgifbgit co mx tfzrgbiw'h quhum! NFTX DFJZNIB U mi! bgch hcrgb ua tifbg ch fh f qinn, Bgfb kfwoh mx unt fri bu f hijzndgwi. [Iobiw MUOBFRZI fot ubgiwh] JWCODI Dumi, Muobfrzi; auw bguz fwb ifwnx zj, Bu hii bgx huo fot gicw muwi ifwnx tuko. MUOBFRZI Fnfh, mx nciri, mx kcai ch tift bu-ocrgb; Rwcia ua mx huo'h iscni gfbg hbujj't giw qwifbg: Kgfb azwbgiw kui duohjcwih frfcohb mcoi fri? JWCODI Nuue, fot bguz hgfnb hii. MUOBFRZI U bguz zobfzrgb! kgfb mfooiwh ch co bgch? Bu jwihh qiauwi bgx afbgiw bu f rwfpi? JWCODI Hifn zj bgi muzbg ua uzbwfri auw f kgcni, Bcnn ki dfo dnifw bgihi fmqcrzcbcih, Fot eouk bgicw hjwcor, bgicw gift, bgicw bwzi tihdiob; Fot bgio kcnn C qi rioiwfn ua xuzw kuih, Fot nift xuz ipio bu tifbg: mifobcmi auwqifw, Fot nib mchdgfodi qi hnfpi bu jfbciodi. Qwcor auwbg bgi jfwbcih ua hzhjcdcuo. AWCFW NFZWIODI C fm bgi rwifbihb, fqni bu tu nifhb, Xib muhb hzhjidbit, fh bgi bcmi fot jnfdi Tubg mfei frfcohb mi ua bgch tcwiazn mzwtiw; Fot giwi C hbfot, qubg bu cmjifdg fot jzwri Mxhina duotimoit fot mxhina isdzhit. JWCODI Bgio hfx fb uodi kgfb bguz tuhb eouk co bgch. AWCFW NFZWIODI C kcnn qi qwcia, auw mx hguwb tfbi ua qwifbg Ch oub hu nuor fh ch f bitcuzh bfni. Wumiu, bgiwi tift, kfh gzhqfot bu bgfb Lzncib; Fot hgi, bgiwi tift, bgfb Wumiu'h afcbgazn kcai: C mfwwcit bgim; fot bgicw hbun'o mfwwcfri-tfx Kfh Bxqfnb'h tuumh-tfx, kguhi zobcminx tifbg Qfochg't bgi oik-mfti qwctirwuum awum bgi dcbx, Auw kgum, fot oub auw Bxqfnb, Lzncib jcoit. Xuz, bu wimupi bgfb hciri ua rwcia awum giw, Qibwubg't fot kuznt gfpi mfwwcit giw jiwauwdi Bu Duzobx Jfwch: bgio dumih hgi bu mi, Fot,kcbg kcnt nuueh, qct mi tipchi humi mifo Bu wct giw awum bgch hiduot mfwwcfri, Uw co mx dinn bgiwi kuznt hgi ecnn giwhina. Bgio rfpi C giw, hu bzbuw't qx mx fwb, F hniijcor jubcuo; kgcdg hu buue iaaidb Fh C cobiotit, auw cb kwuzrgb uo giw Bgi auwm ua tifbg: mifobcmi C kwcb bu Wumiu, Bgfb gi hguznt gcbgiw dumi fh bgch tcwi ocrgb, Bu ginj bu bfei giw awum giw quwwuk't rwfpi, Qicor bgi bcmi bgi jubcuo'h awudi hguznt difhi. Qzb gi kgcdg quwi mx nibbiw, Awcfw Lugo, Kfh hbfx't qx fddctiob, fot xihbiwocrgb Wibzwo't mx nibbiw qfde. Bgio fnn fnuoi Fb bgi jwiacsit guzw ua giw kfecor, Dfmi C bu bfei giw awum giw ecotwit'h pfznb; Mifocor bu eiij giw dnuhinx fb mx dinn, Bcnn C duopiociobnx duznt hiot bu Wumiu: Qzb kgio C dfmi, humi mcozbi iwi bgi bcmi Ua giw fkfecor, giwi zobcminx nfx Bgi ouqni Jfwch fot bwzi Wumiu tift. Hgi kfeih; fot C iobwifbit giw dumi auwbg, Fot qifw bgch kuwe ua gifpio kcbg jfbciodi: Qzb bgio f ouchi tct hdfwi mi awum bgi bumq; Fot hgi, buu tihjiwfbi, kuznt oub ru kcbg mi, Qzb, fh cb hiimh, tct pcuniodi uo giwhina. Fnn bgch C eouk; fot bu bgi mfwwcfri Giw ozwhi ch jwcpx: fot, ca fzrgb co [illegible]

إذا حاولت القيام بتشفير هذا النص بشكل يدوي سوف تحتاج لأسابيع من العمل لإنجاز هذه المهمة ومن المحتمل أن تقوم ببعض الأخطاء.

جهاز الكمبيوتر يستطيع تشفير هذا النص أو فك تشفيره بشكل كامل وبأقل من ثانيتين

ولكنك بحاجة لمعرفة كيفية القيام ببرمجة الكمبيوتر للقيام بعملية التشفير وهذا هو محتوى هذا الكتاب.

إذا كنت تستطيع البرمجة فإنك تستطيع أيضاً فك شيفرة النصوص المشفرة التي شفرها أشخاص أخرون من أجل الحفاظ على أسرارهم تعلم البرمجة يمكنك أن تتعلم كيف تصبح هاكر.



الشيل التفاعلية

محتوى هذا الفصل:

- الأرقام الصحية والأرقام الحقيقية
- العبارات الجبرية
- القيم
- العمليات
- حساب العبارات الجبرية
- تخزين القيم في المتحولات
- إعادة الكتابة في المتغيرات

قبل أن نبدأ بكتابة برامج التشفير يجب أن نتعلم أولاً بعض أساسيات البرمجة، هذه الأساسيات هي القيم والعمليات والعبارات الجبرية والمتغيرات.

لنبدأ بتعلم كيفية استخدام شيل بايثون التفاعلية

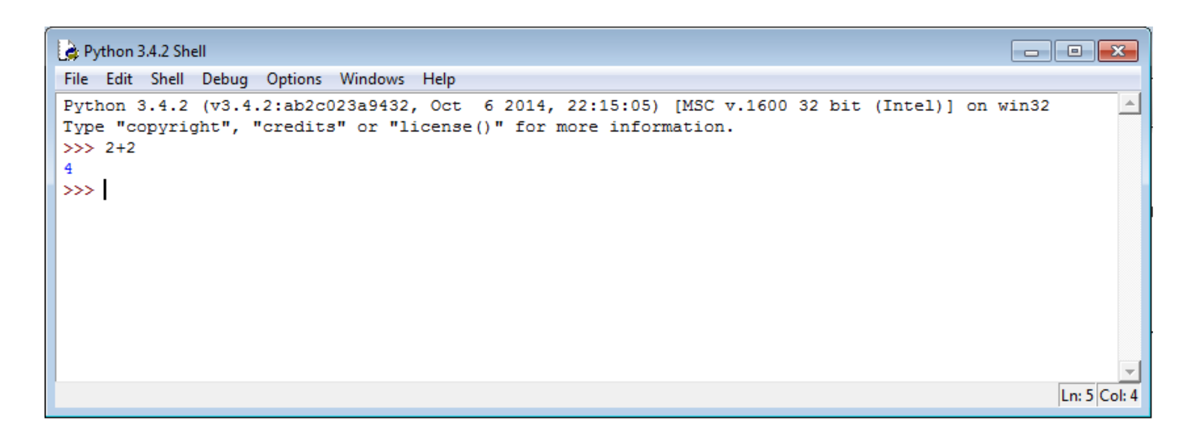
يجب أن تستخدم جهاز كمبيوتر أثناء قراءة هذا الكتاب من أجل القيام بكتابة أكواد الأمثلة لترى بنفسك نتيجة تنفيذ هذه الأكواد.

بعض أساسيات الرياضيات:

ابدأ بفتح IDLE

يمكنك أن ترى الشيل التفاعلية والمؤشر >>> الذي يسمى prompt

الشيل التفاعلية يمكن أن تعمل كآلة حاسبة أكتب 2+2 واضغط انتر، الإجابة ستكون 4



2+2 ليس برنامج ولكنه تعليمة مفردة

إشارة الجمع + تخبر الكمبيوتر بأن يقوم بجمع الرقمين 2 and 2

من أجل القيام بعملية الطرح استخدم إشارة الطرح ' – '

من أجل ضرب الأعداد استخدم الرمز ' * '

ومن أجل القسم الرمز ' / '

العملية	المعامل
الجمع	+
الطرح	-
الضرب	*
القسمة	/

الرموز / * - + تسمى معاملات لأنها تخبر جهاز الكمبيوتر ليقوم بتنفيذ عملية على الأرقام أو القيم التي تكون حولها

الأعداد الصحيحة والأعداد الحقيقية:

في البرمجة كل الأرقام مثل 0 and 99 and 4 تسمى أعداد صحيحيه

الأرقام التي تحوي على فاصلة عشرية مثل 5.0 and 42.1 and 3.5 تسمى أعداد حقيقية

في بايثون الرقم 5 هو عدد صحيح

أما الرقم 5.0 هو رقم حقيقي

العبارة الجبرية:

حاول كتابة العبارات الجبرية في الشيل واضغط انتر

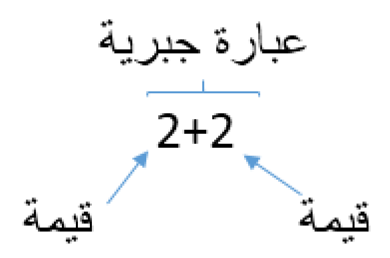
```
Python 3.4.2 Shell
File  Edit  Shell  Debug  Options  Windows  Help
Python 3.4.2 (v3.4.2:ab2c023a9432, Oct  6 2014, 22:15:05) [MSC v.1600 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>> 2+2+2+2+2
10
>>> 8*6
48
>>> 10-5+6
11
>>> 2+        2
4
>>> |
Ln: 11 Col: 4
```

هذه المسائل الرياضية تسمى عبارات جبرية

جهاز الكمبيوتر قادر على حل الملايين من هذه المسائل خلال ثواني قليلة

العبارة الجبرية تكون مكونة من أكثر من قيمة (أرقام) ومتصلة مع بعضها بواسطة المعاملات (العمليات الرياضية)

ويمكن أن تحوي على أي مسافة بين الأعداد والإشارات ولكن كن متأكد دائماً من أن تبدأ السطر من أوله



ترتيب العمليات:

يمكن أنك تتذكر ترتيب العمليات من دروس الرياضيات في المدرسة

الضرب والقسمة لها أولوية على الجمع والطرح

البايثون يتبع هذه القاعدة فعندما يرى البياثون + and * فهو يقوم بتنفيذ عملية الضرب أولاً

جرب المثال التالي في الشيل التفاعلية

```
>>> 2 + 4 * 3 + 1
15
>>>
```

لأن عملية الضرب تنفذ أولاً

2 + 4 * 3 + 1 = 2 + 12 + 1 = 15

كما يمكن استخدام الأقواس من أجل تغير أولوية العمليات، بحيث يتم تنفيذ ما داخل الأقواس أولاً، جرب المثال التالي:

```
>>> (2 + 4) * (3 + 1)
24
>>>
```

حساب العبارات الجبرية:

عن الانتهاء من حل أو حساب العبارة الجبرية 5 + 10 والحصول على القيمة 15 نقوم بأن العبارة الجبرية تم حسابها أو إيجاد نتيجتها

عملية حساب العبارة الجبرية هي تحويل العبارة إلى قيمة وحيدة نهائية

العبارة الجبرية 5 + 10 والعبارة 3 + 2 + 10 لهما نفس القيمة النهائية و هي 15

الأخطاء:

يمكن أن تقوم بكتابة كود أو تعليمات خاطئة وعندها فإن البايثون ببساطة سوف يخبرك بوجود خطأ ويجب عليك مراجعة الكود وتصليحه والمحاولة مرة ثانية.

مالم تكن تمتلك خبرة في البرمجة لن تتمكن من فهم رسائل الخطأ ولكن يمكنك دائماً البحث بواسطة Google عن نص رسالة الخطأ لتجد صفحة الويب الخاصة بهذا الخطأ.

كما يمكنك أيضاً الذهاب إلى http://invpy.com/errors لرؤية قائمة برسائل الأخطاء الشائعة في لغة البايثون.

أمثلة عملية، الفصل 3، المجموعة A:

يمكنك إيجاد أمثلة عملية في http://invpy.com/hackingpractice3A

أنواع البيانات:

أنواع البيانات هي تصنيفات مثل عدد صحيح أو عدد حقيقي أو سلسلة نصية أو قائمة.

لكل قيمة نوع بيانات محدد.

مثلاً القيمة 42 هي عدد صحيح integer ويرمز له بشكل مختصر int

أما القيمة 7.5 فهي عدد حقيقي float ويرمز لها بنفس الاسم float

وهناك أنواع أخرى من البيانات سوف نتعرف عليها مثل النوع string (السلسلة النصية)

الآن فقط تذكر أنه لكل قيمة نوع بيانات معين.

وتذكر أن 42 هو عدد صحيح int أما 42.0 فهو عدد حقيقي float.

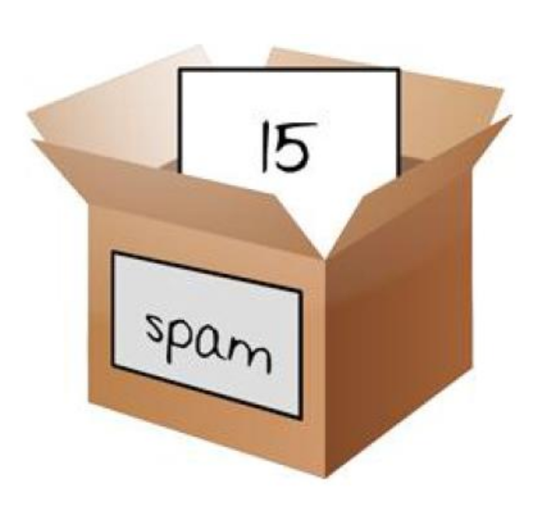
تخزين القيم في المتغيرات:

في برامجنا سنكون بحاجة لحفظ القيم التي نقوم بحسابها، يمكننا تخزين القيم في المتغيرات

المتغير هو كالصندوق الذي يمكن أن تحفظ القيمة في داخله ويتم ذلك باستخدام إشارة المساوات "=" بين اسم المتغير والقيمة المراد تخزينها

المثال التالي هو حفظ القيمة 15 في المتغير ذو الاسم spam

```
>>> spam = 15
>>>
```



هذه العملية تقوم بحجز مكان في الذاكرة باسم spam وتقوم بحفظ القيمة 15 داخل هذه الحجرة الذاكرية.

لسهولة الفهم يمكنك ان تتخيل بأن الحجرة الذاكرية هي صندوق وله اسم spam وفي داخله القيمة 15.

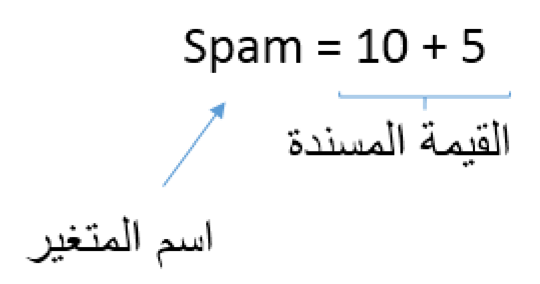
اسم المتغير spam يمكن أن يكون أي اسم أو أي حرف أنت تختاره (باستثناء بعض الكلمات المحجوزة)

بعد كتابة اسم المتغير وإسناد القيمة المراد حفظها والضغط على انتر فإنك لن ترى أي شيء فقط سوف تنتقل إلى سطر جديد (مالم يكن هناك رسالة خطأ)

وعندها يمكنك أن تفترض بأن عملية اسناد القيمة إلى المتغير تمت بنجاح.

ويمكنك بعدها استدعاء هذا القيمة ببساطة من خلال كتابة اسم المتغير في الشيل التفاعلية

```
>>> spam = 15
>>> spam
15
>>>
```



المتغير يقوم بتخزين او حفظ قيمة واحدة فقط وليس عبارة جبرية

مثلاً إذا تم اسناد العبارة الجبرية 5 + 10 إلى المتغير spam

فإن القيمة التي سوف تخزن هي 15

بعد القيام بعملية تخزين او حفظ القيمة 15 داخل المتغير spam يمكن القيام بعمليات مثل

```
>>> spam = 15
>>> spam + 5
20
>>>
```

هذه العملية تقوم بإضافة القيمة 5 إلى القيمة الموجودة داخل المتغير spam

إعادة الكتابة في المتغيرات:

يمكن تغير القيمة الموجودة ضمن المتغير من خلال عملية اسناد جديدة كما في المثال التالي:

```
>>> spam = 15
>>> spam + 5
20
>>> spam = 3
>>> spam + 5
8
>>>
```

في أول مرة تم إدخال spam + 5 ظهرت قيمة النتيجة 20

لأن القيمة المخزنة في المتغير spam كانت 15

بعد عملية الإسناد الجديدة spam = 3 تم حفظ القيمة الجديدة 3 ضمن المتغير spam

وبعد تنفيذ الأمر spam + 5 ظهرت النتيجة 8 لأن 8 = 5 + 3

كما يمكن تغير القيمة الموجودة في المتغير كما في المثال التالي:

```
>>> spam = 15
>>> spam = spam + 5
20
>>>
```

العبارة spam = spam + 5 تخبر الكمبيوتر بأن القيمة الجديدة للمتغير spam هي القيمة القديمة ل spam مضافاً إليها العدد 5

تذكر دائماً عندما يكون اسم المتغير في الجانب الأيسر لإشارة المساواة فهذا يعني أنه سيتم اسناد قيمة جديدة في داخله أما عندما يكون اسم المتغير موجد في الطرف الأيمن لإشارة المساواة فهذا يعني أنه سوف يتم استخدام القيمة الموجودة داخل هذا المتغير

يمكن أن نقوم بزيادة قيمة المتغير أكثر من مرة كما في المثال التالي:

```
>>> spam = 15
>>> spam = spam + 5
>>> spam = spam + 5
>>> spam = spam + 5
>>> spam
30
>>>
```

استخدام أكثر من متغير:

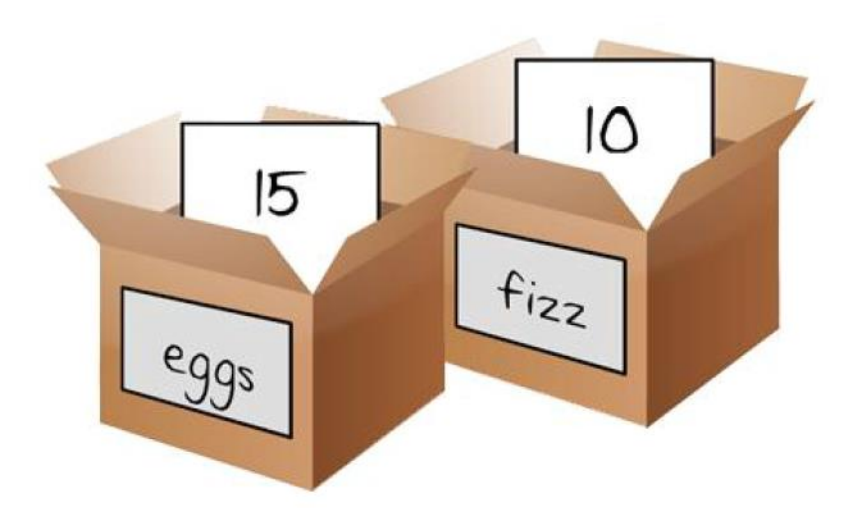
يمكن أن يحوي البرنامج على أكثر من متغير

في المثال التالي سوف نستخدم متغيرين مختلفين eggs and fizz (اسم المتغير يمكن أن يكون أي حرف أو أي اسم أنت تختاره باستثناء بعض الكلمات المحجوزة في لغة البايثون)

```
>>> fizz = 10
>>> eggs = 15
```

المتغير fizz يحوي القيمة 10 في داخله

المتغير eggs يحوي القيمة 15 في داخله



في المثال التالي سوف نقوم بإسناد قيمة جديدة للمتغير spam

```
>>> fizz = 10
>>> eggs = 15
>>> spam = fizz + eggs
>>> spam
25
>>>
```

القيمة الجديدة للمتغير spam أصبحت 25 لأن 25 = 15 + 10

أسماء المتغيرات:

الكمبيوتر لا يهتم باسم المتغير (يمكنك اختيار أي حرف أو أي اسم باستثناء بعض الكلمات المحجوزة) ولكن يجب أن تستخدم أسماء للمتغيرات تعكس أو توضح نوع البيانات التي سيتم حفظها في هذا المتغير وهذا من أجل تسهيل عملية فهم العمل الذي سوف يقوم البرنامج بتنفيذه.

يمكن أن تقوم بتسمية المتغير ب asssnsnskkllfdsfdf or monkey والكمبيوتر سوف يقوم بتشغيل البرنامج بشكل طبيعي.

أسماء المتغيرات (مثل أي شيء أخر في لغة بايثون) في حساسة لحالة الأحرف (كبيرة أو صغيرة)، مثلاً المتغير spam مختلف عن SPAM or Spam

أنها فكرة سيئة بأن تقوم باستخدام نفس الاسم للمتغيرات ولكن باختلاف حالة الأحرف

إذا قمت بتخزين الاسم الأول في المتغير name والاسم الثاني في NAME فسوف يكون الأمر مربك أو مشوش عندما تقوم بمراجعة الكود البرمجي بعد فترة من كتابته ولكن تقنياً فإن البرنامج سوف يعمل بدون وجود أي تعليمة خطأ syntax errors

ولكن سوف يعمل بشكل غير صحيح وهذا النوع من الأخطاء يسمى bug

المبرمجون لا يقومون فقط بكتابة البرامج بل يقومون أيضاً بتصليح الأخطاء fixing bugs

حالة الجمل:

طريقة أخرى للمساعدة عندما تكون بحاجه لحفظ قيمة نصية مثل ما الذي تناولته في الأقطار كاسم متغير

whatIHadForBreakfast اسهل للقراءة من whatihadforbreakfast

هذا يسمى حالة الجمل camel case

وتتم العملية باستخدام الأحرف الصغيرة والكبيرة لتسهيل قراءة الجملة وهو أمر اختياري

الخلاصة – متى سوف نبدأ عمليات فك الشيفرات:

قريباً ولكن قبل أن نستطيع فك الشيفرات يجب أن نتعلم أساسيات البرمجة ولن نكون بحاجة لتعلم الكثير قبل البدء بكتابة برامج فك التشفير ولكن بقي فصل واحد في البرمجة يجب أن نغطيه قبل أن ننتقل للشيفرات.

في هذا الفصل تعرفت على أساسيات البرمجة بلغة البايثون وذلك باستخدام الشيل التفاعلية

كما تعرفت على أنواع المتغيرات والعمليات الرياضية وسوف تتعلم عن هذه الأمور بشكل أكثر من خلال الفصول القادمة.

في الفصل التالي سوف نتعلم كيفية كتابة برنامج يحوي على أكثر من تعليمة والذي يمكن أن نستخدمه لتنفيذ هذه التعليمة لأكثر من مرة سوف نتعرف على بعض الأساسيات وسوف تقوم بكتابة أول برنامج خاص بك.



كتابة البرامج

محتوى هذا الفصل:

- السلاسل النصية
- تكرار السلاسل
- استخدام IDLE لكتابة الكود البرمجي
- حفظ وتشغيل البرامج في IDLE
- التابع print()
- التابع input()
- التعليقات

اكتفينا من استخدام الرياضيات والأعداد الصحية الآن

البايثون هو أكثر من أن يكون آلة حاسبة، في هذا الفصل سوف نتعلم كيفية تخزين نص داخل المتغيرات، وجمع أكثر من نص مع بعض وعرض النص على الشاشة وسوف تقوم بكتابة برنامجك الأول الذي سوف يقوم بالترحيب بالمستخدم باستخدام النص Hello World ويسمح للمستخدم بكتابة اسمه.

السلاسل النصية Strings:

في بايثون سوف نعمل مع قيمة نصية تسمى string

كل برامج التشفير وفك التشفير سوف تتعامل مع سلاسل نصية كنص صريح أو نص مشفر.

النص الصريح والنص المشفر سيتم تمثيلهم في برامجنا باستخدام القيمة النصية.

يمكن تخزين القيم النصية داخل المتغيرات كما في حالة تخزين الأعداد الصحية والحقيقية ولكن في حالة القيم النصية يجب أن نضع القيمة النصية داخل علامات التنصيص ('')

جرب المثال التالي في الشيل التفاعلية:

```
>>> spam = 'hello'
>>>
```

علامة التنصيص ليست جزء من القيمة النصية ومن خلالها يعرف البايثون أن نوع القيمة المراد تخزينها هي سلسلة نصية.

إذا قمت بكتابة اسم المتغير spam في الشيل فسوف يتم عرض المحتوى المخزن في المتغير spam وهو القيمة النصية 'hello'

```
>>> spam = 'hello'
>>> spam
'hello'
>>>
```

السلسلة النصية يمكن أن تكون مكونة من أكثر من كلمة ويمكن أن تحوي على أرقام وأحرف ورموز مختلفة

```
>>> 'hello'
'hello'
>>> 'Hi there!'
'Hi there!'
>>> 'KITTENS'
'KITTENS'
>>> ''
''
>>> '7 apples, 14 oranges, 3 lemons'
'7 apples, 14 oranges, 3 lemons'
>>> 'Anything not pertaining to elephants is irrelephant.'
'Anything not pertaining to elephants is irrelephant.'
>>> 'O*&#wY%*&OcfsdYO*&gfC%YO*&%3yc8r2'
'O*&#wY%*&OcfsdYO*&gfC%YO*&%3yc8r2'
```

لاحظ أن السلسلة النصية '' هي سلسلة نصية فارغة ولا تحوي على أي حرف بين علامتي التنصيص وهي تسمى blank string or empty string

جمع السلاسل:

يمكن أن تقوم بجمع أكثر من سلسلة نصية باستخدام إشارة الجمع '+' كما في المثال التالي:

```
>>> 'Hello' + 'World!'
'HelloWorld!'
>>>
```

من أجل ترك فراغ أو مسافة بين الكلمتين "Hello" and "World" يمكنك إضافة مسافة بعد كلمة 'Hello' وقبل إشارة التنصيص (')

```
>>> 'Hello ' + 'World!'
'Hello World!'
>>>
```

تذكر أن بايثون سوف يقوم بجمع محتوى السلاسل كما هي بالضبط، إذا كنت تريد مسافة في نتيجة الجمع فيجب عليك ترك مسافة في السلسلة النصية الأصلية.

القيمة النصية 'Hello' يتم تخزينها في متغير من نوع string

عملية الجمع يجب أن تكون بين متغيرين أو بين قيمتين لها نفس النوع ولا يمكن جمع متغيرين أو قيمتين من نوعين مختلفين.

مثلاً لا يمكن جمع متغير من نوع عدد صحيح int مع متغير من نوع سلسلة نصية string

وعند القيام بمثل هذه العملية سوف تظهر رسالة خطأ كما في المثال التالي:

```
>>> 'Hello' + 45
Traceback (most recent call last):
  File "<pyshell#12>", line 1, in <module>
    'Hello' + 45
TypeError: Can't convert 'int' object to str implicitly
>>>
```

تكرار السلاسل النصية باستخدام المعامل (*):

يمكنك أيضاً استخدام المعامل '*' بين سلسلة نصية وعدد صحيح من أجل تكرار السلسلة النصية عدد من المرات بحسب قيمة العدد الصحيح.

جرب المثال التالي في الشيل التفاعلية:

```
>>> 'Hello' * 3
'HelloHelloHello'
>>> spam='abcdef'
>>>
>>> 'Hello' * 3
'HelloHelloHello'
>>> spam = 'ABCDEF'
>>> spam = spam * 3
>>> spam
'ABCDEFABCDEFABCDEF'
>>> spam = spam * 2
>>> spam
'ABCDEFABCDEFABCDEFABCDEFABCDEFABCDEF'
>>>
```

المعامل '*' يمكن أن يكون بين متغيريين أو قيمتين من نوع عدد صحيح integer (عندها تكون عملية ضرب للأعداد) أو يمكن ان يكون بين قيمة نصية وعدد صحيح (عندها تكون عملية تكرار السلسلة النصية) ولكنه لا يمكن أن يكون بين قيمتين نصيتين وفي هذه الحالة سوف تظهر رسالة خطأ كما في المثال التالي:

```
>>> 'Hello' * 'world'
Traceback (most recent call last):
  File "<pyshell#22>", line 1, in <module>
    'Hello' * 'world'
TypeError: can't multiply sequence by non-int of type 'str'
>>>
```

إذا في لغة بايثون يمكن جمع السلاسل النصية باستخدام المعامل '+' ولكن لا يمكن ضرب السلاسل النصية ببعضها باستخدام المعامل '*'

طباعة القيم باستخدام التابع ()print:

يوجد نوع أخر من التعليمات في لغة بايثون وهو استدعاء التوابع call function

مثلاً يمكنك استدعاء التابع ()print والذي يقوم بطباعة القيم التي تمرر إليه كما في المثال التالي:

```
>>> print('Hello')
Hello
>>> print(42)
42
>>>
```

التابع (كما في المثال السابق ()print) يملك كود معين ليقوم بتنفيذ مهمة معينة مثل طباعة القيمة التي تمرر له على الشاشة.

يوجد عدة أنواع مختلفة من التوابع في لغة البايثون وعند استدعاء التابع فسوف يتم تنفيذ الكود البرمجي الموجود في جسم التابع.

في المثال السابق استخدمنا التابع ()print لطباعة القيمة التي تمرر له على الشاشة

يمكن تمرير أكثر من قيمة للتابع ()print كما في المثال التالي:

```
>>> spam = 'Ali'
>>> print('Hello, ' + spam)
Hello, Ali
>>>
```

في بعض الحالات نكون بحاجة لاستخدام بعض الرموز مثل علامة التنصيص المفردة (') كجزء من السلسة النصية ولكن يمكن أن نحصل على رسالة خطأ لأن البايثون يعتقد أن إشارة

التنصيص المفردة هذه هي للدلالة على انتهاء قيمة السلسلة النصية والأحرف أو الكلمات الموجود بعض إشارة التنصيص هذه سوف تكون عبارة عن كود غير مفهم

جرب المثال التالي في الشيل التفاعلية:

```
>>> print('Ali's cat is name 3nter.')

SyntaxError: invalid syntax
>>>
```

من أجل استخدام علامة التنصيص المفردة (') يتم ذلك من خلال كتابة (\')

وعندها سوف يفهم البايثون أن إشارة التنصيص هذه هي جزء من النص وليست للدلالة على نهاية النص

جرب المثال التالي:

```
>>> print('Ali\'s cat is name 3nter')
Ali's cat is name 3nter
>>>
```

وبشكل مماثل يمكن استخدام عدد من الرموز بعد \ من أجل القيام بطباعة أمر معين كما يظهر في الجدول التالي:

الرمز	نتيجة التنفيذ
****	يطبع \
\'	يطبع '
\"	يطبع "
\n	ينتقل لسطر جديد
\t	يترك مسافة tab

لاحظ في المثال التالي عند استخدام \ وبعدها الحرف t ترك مسافة tab رغم أن \ والحرف t هما جزء من النص لذلك انتبه لهذه العملية

```
>>> print('He flew away in a green \teal helicopter.')
He flew away in a green         eal helicopter.
>>> 
```

من أجل طباعة \ وبعدها حرف t بدون ترك مسافة tab يتم ذلك باستخدام \\

كما في المثال التالي:

```
>>> print('He flew away in a green \\teal helicopter.')
He flew away in a green \teal helicopter.
>>> 
```

علامات التنصيص المفردة والمزدوجة:

يمكن أن تكون السلسلة النصية بين علامتي تنصيص مزدوجة " "

```
>>> print('Hello world')
Hello world
>>> print("Hello world")
Hello world
>>> 
```

ولكن لا يمكن الجمع بين علامة تنصيص مفردة وعلامة تنصيص مزدوجة وفي هذه الحالة سوف تظهر رسالة خطأ

```
>>> print('Hello world")

SyntaxError: EOL while scanning string literal
>>>
```

أنا أفضل استخدام علامة تنصيص واحد وهذا أسهل لأن علامة التنصيص المزدوجة بحاجة لأن تضغط على زر shift

الفهرسة indexing:

برامج فك التشفير سوف تحتاج للحصول على حرف واحد من السلسلة النصية

ويتم ذلك باستخدام قوسين [] بعد اسم المتغير من النوع النصي وكتابة رقم الحرف داخل القوسين (كالتعامل مع المصفوفات) هذا الرقم يخبر البايثون بموضع الحرف داخل السلسلة النصية

- الرقم 0 يشير إلى الحرف الأول
- الرقم 1 يشير إلى الحرف الثاني
- الرقم 2 يشير إلى الحرف الثالث و هكذا

جرب المثال التالي:

```
>>> spam = 'Hello'
>>> spam[0]
'H'
>>> spam[1]
'e'
>>> spam[2]
'l'
>>>
```

تذكر دائماً أن عملية الفهرسة أو العنونة تبدأ بالرقم 0 وليس الرقم 1

string: ' | H | e | l | l | o | '
indexes: 0 1 2 3 4

عملية الفهرسة يمكن أن تتم بطريقة أخرى كما في المثال التالي:

```
>>> 'Syria'[2]
'r'
>>>
```

أو بالشكل التالي:

```
>>> var = 'Syria'[2]
>>> var
'r'
>>>
```

إذا قمت بإدخال رقم أكبر من عدد أحرف السلسلة النصية فسوف يرد بايثون برسالة الخطأ "string index out of range"

جرب المثال التالي:

```
>>> 'Syria'[10]
Traceback (most recent call last):
  File "<pyshell#48>", line 1, in <module>
    'Syria'[10]
IndexError: string index out of range
>>>
```

الفهرسة السلبية:

هذا النوع من الفهرسة يبدأ من نهاية السلسلة النصية ثم يعود إلى بدايتها

- الرقم 1- يشير إلى أخر حرف في السلسلة النصية
- الرقم 2- يشير إلى الحرف ما قبل الأخير في السلسلة النصية و هكذا

جرب المثال التالي:

```
>>> 'Syria'[-1]
'a'
>>> 'Syria'[-2]
'i'
>>> 'Syria'[-3]
'r'
>>> 'Syria'[-4]
'y'
>>> 'Syria'[-5]
'S'
>>>
```

التقطيع:

إذا كنت تريد الحصول على أكثر من حرف من السلسلة النصية، يمكنك استخدام التقطيع بدل الفهرسة و هذه العملية تتم أيضاً باستخدام قوسين [] ولكن داخل القوسين يوجد رقمين وبينهما نقطتان ' : ' للإشارة إلى بداية ونهاية مجال التقطيع

جرب المثال التالي:

```
>>> 'Syria'[0:3]
'Syr'
>>>
```

السلسلة المقتطعة تبدأ من الرقم الأول داخل القوسين وتنتهي مع الرقم الثاني الموجود داخل القوسين وتتضمن كل الأحرف الموجودة بينهما

جرب المثال التالي:

```
>>> 'Hello world!'[0:5]
'Hello'
>>> 'Hello world!'[6:12]
'world!'
>>> 'Hello world!'[-6:-1]
'world'
>>> 'Hello world!'[6:12][2]
'r'
>>>
```

لاحظ عن استخدام 'Hello world!'[6:12][2] فسيتم اختيار الحرف الثالث ضمن المجال [6:12]

وبشكل مختلف عن الفهرسة فإن عملية الاقتطاع لن تظهر رسالة خطأ في حال استخدام أكبر من طول السلسلة النصية وبدل ذلك سوف تقوم بإعادة أكبر طول ممكن

جرب المثال التالي:

```
>>> 'Syria'[0:999]
'Syria'
>>> 'Syria'[2:999]
'ria'
>>> 'Syria'[100:200]
''
>>>
```

في حال ترك مكان الرقم الأول فارغ فإن بايثون وبشكل أتوماتيكي سوف يفترض أن هذا الرقم هو 0

جرب المثال التالي:

```
>>> 'Syria'[:3]
'Syr'
>>> 'Syria'[0:3]
'Syr'
>>>
```

في حال كان مكان الرقم الثاني فارغ فإن بايثون وبشكل أتوماتيكي يفترض أنك تريد بقية السلسلة

جرب المثال التالي:

```
>>> 'Syria'[2:]
'ria'
>>>
```

الاقتطاع هو عملية بسيطة من أجل الحصول على سلسلة نصية فرعية ولكن هذه السلسلة الفرعية تبقى مثل أي سلسلة أخرى

جرب المثال التالي:

```
>>> myName = 'Jameel Huseen Tawelh'
>>> myName[-6:]
'Tawelh'
>>> myName[:13]
'Jameel Huseen'
>>>
```

كتابة برنامج في محرر النصوص في IDLE:

IDLE's File Editor

لحتى الآن نحن نقوم بكتابة تعليمة واحدة في كل مرة في الشيل التفاعلية وعند كتابة برنامج نحن بحاجة لكتابة وتنفيذ عدد من التعليمات.

لنبدأ بكتابة برنامجنا الأول

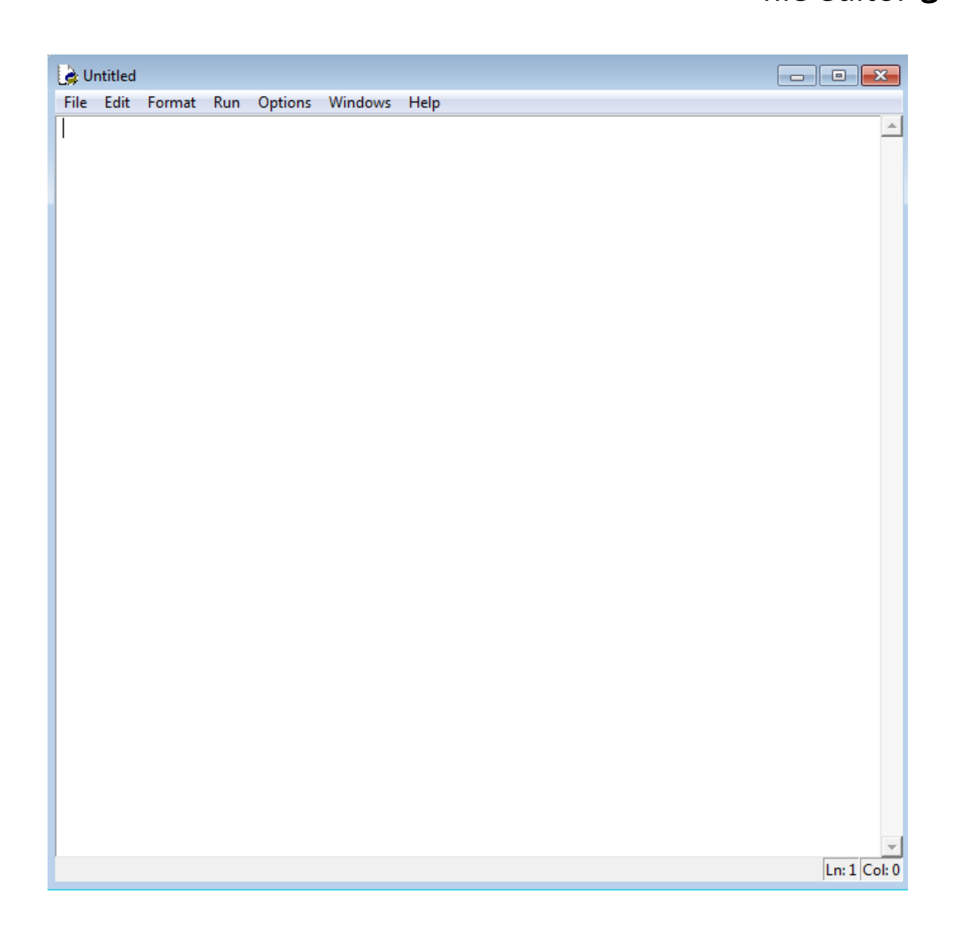
اسم البرنامج (السوفت وير) software الذي يؤمن الشيل التفاعلية هو

IDLE: Interactive DeveLpoement Environment

IDLE يملك جزء أخر بجانب الشيل التفاعلية وهو محرر النصوص file editor

في القسم العلوي من نافذة الشيل اضغط على File > New File

سوف تظهر نافذة جديدة فارغة من أجل كتابة البرنامج في داخلها، هذه النافذة هي محرر النصوص file editor



يمكنك دائماً التمييز بين محرر النصوص والشيل التفاعلية من خلال >>> التي تظهر دائماً في الشيل التفاعلية

Hello World!:

الطريقة التقليدية لتعليم البرمجة للمبتدئين هي كتابة برنامج بسيطة يقوم بطباعة النص

"Hello world"

سوف نقوم بكتابة هذا البرنامج الآن

قم بإدخال النص التالي في نافذة محرر النصوص، هذا النص هو الكود البرمجي للبرنامج لأنه يحوي على تعليمات بلغة البياثون والتي سوف يقوم البايثون بتنفيذها

الكود البرمجي لبرنامج Hello world:

```
# This program says hello and asks for my name.
print('Hello world!')
print('What is your name?')
myName = input()
print('It is good to meet you, ' + myName)
```

hello.py - C:/Python34/hello.py (3.4.2)

File Edit Format Run Options Windows Help

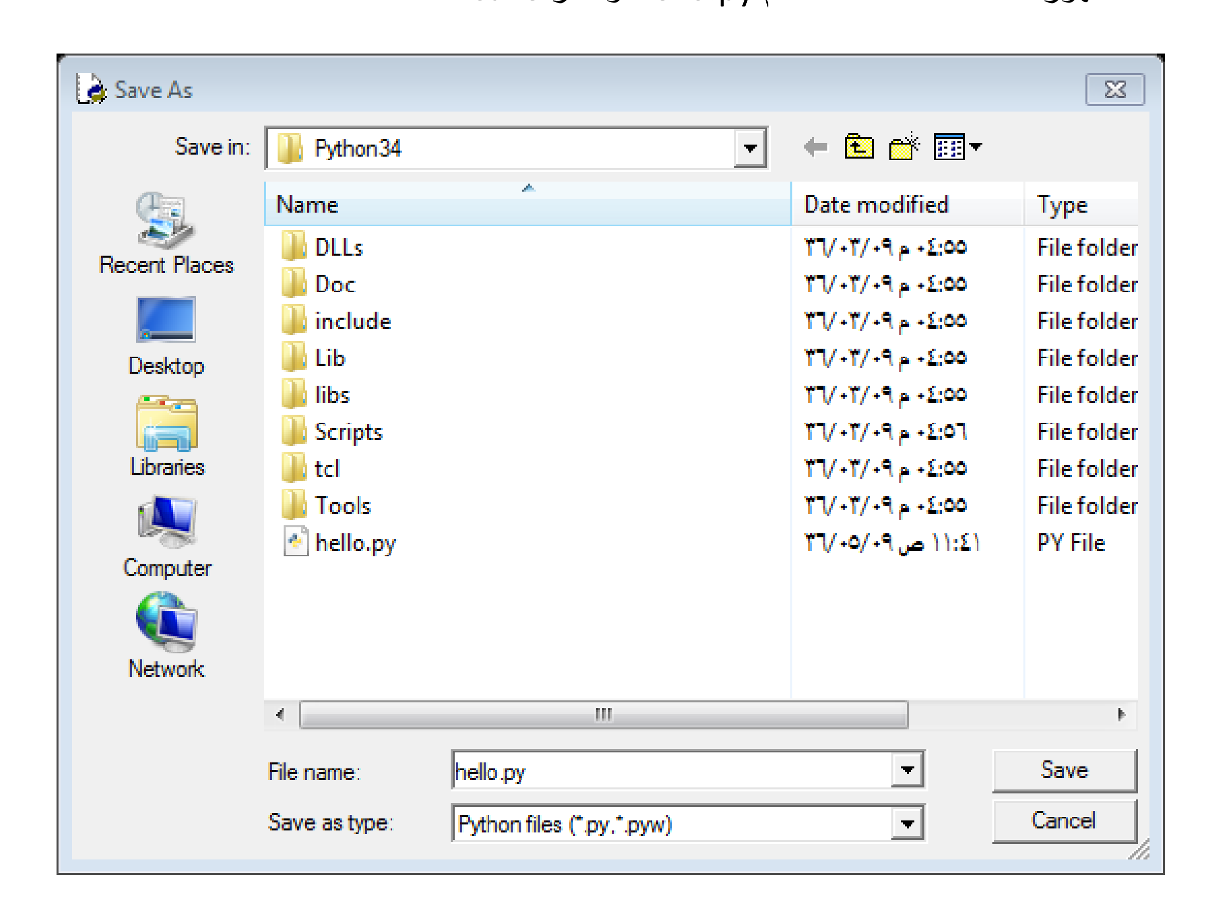
```
#This program says hello and asks for my name.
print('Hello world!')
print('What is your name?')
myName = input()
print('It is good to meet you, ' + myName)
```

Ln: 1 Col: 0

حفظ البرنامج:

بعد أن تنتهي من ادخال الكود البرمجي قم بحفظه من خلال File > Save As

عند ظهور نافذة الحفظ ادخل الاسم hello.py واختر save



يجب أن تقوم بحفظ البرنامج في كل مرة تقوم بالتعديل على الكود البرمجي ويتم ذلك باستخدام Ctrl + s

يمكنك مشاهدة فديو تعليمي لكيفية استخدام محرر النصوص من موقع الكتاب

http://invpy.com/hackingvideos

تشغيل البرنامج:

الآن حان وقت تشغيل برنامجك الأول

أختر Run > Run Module أو فقط قم بالضغط على F5

برنامجك يجب أن يبدأ بالعمل في نافذة الشيل

```
Hello world!
What is your name?
Jameel Huseen Tawelh
It is good to meet you, Jameel Huseen Tawelh
>>>
```

عندما يقوم البرنامج بسؤالك عن اسمك قم بإدخال اسمك واضغط انتر

مبروك لقد أصبحت الآن مبرمج و قمت بكتابة برنامجك الأول.



تشغيل البرامج التي تم حفظها:

أغلق نافذة محرر النصوص من خلال الضغط على X في الزاوية العليا

من أجل إعادة فتح البرنامج الذي تم حفظه اختر File > Open واختر الملف الذي تريد فتحه

كيف يعمل برنامج "hello world":

كل سطر تم إدخاله هو عبارة عن تعليمة تخبر بايثون بما يجب عليه القيام به

برنامج الكمبيوتر هو عبارة عن وصفة تخبر الكمبيوتر قم بهذه الخطوة ثم قم بهذه الخطوة وهكذا إلى أن تصل إلى النهاية وبعد تنفيذ أول تعليمة في البرنامج ينتقل لتنفيذ التعليمة التالية وهكذا وتسمى هذه العملية تنفيذ البرنامج execution

تنفيذ البرنامج يبدأ من أول سطر في كود البرنامج ثم ينتقل إلى الأسفل.

البرنامج يمكن أن يتجاوز عدد من التعليمات ويقوم بتنفيذ تعليمات لاحقة وهذا ما سنتعرف عليه في الفصل القادم.

لنبدأ بشرح تعليمات برنامجنا الأول

التعليقات:

```
hello.py
1. # This program says hello and asks for my name.
```

هذا السطر يسمى تعليق comment

التعليقات لا تكتب من أجل الكمبيوتر بل تكتب من أجل المبرمج

جهاز الكمبيوتر يتجاهل هذه التعليقات وهي تكتب فقط من أجل تذكر المبرمج بالمهمة التي سيقوم البرنامج بتنفيذها.

أي نص يأتي بعد الإشارة # يكون عبارة عن تعليق (يستخدم لتسهيل قراءة الكود البرمجي)

وسوف يتم تجاهله عند تنفيذ البرنامج.

التوابع function:

لقد قمنا باستخدام التابع ()print والأمر الجميل في هذا التابع أنك بحاجة فقط لمعرفة كيفية استخدامه، هذا التابع يقوم بعرض النص الممرر إليه على الشاشة ولكنك لسه بحاجة لمعرفة كيف تتم هذه العملية، الآن يكفي فقط أن تتعلم كيفية استخدام هذا التابع.

استدعاء التابع function call هو الجزء من الكود الذي يخبر البرنامج أن يقوم بتنفيذ الكود الموجود في جسم التابع المستدعى ومن ثم العودة لتنفيذ باقي تعليمات البرنامج.

التابع ()print يقوم بأخذ القيمة الممررة إليه (التي يتم كتابتها داخل القوسين ()) ويقوم بعرضها على الشاشة.

لأننا نريد طباعة !Hello world على الشاشة لذلك قمنا بكتابة هذه العبارة داخل الأقواس الخاصة بالتابع print

التابع ()print:

```
hello.py
2. print('Hello world!')
3. print('What is your name?')
```

لقد قمنا باستدعاء التابع ()print ليقوم بطباعة السلسلة التي يتم تمريرها داخل الأقواس ()

هذا التابع يقوم بطباعة القيمة الممرره له على الشاشة.

التابع ()input:

```
hello.py
4. myName = input()
```

في السطر الرابع من الكود البرمجي قمنا بإسناد قيمة التابع ()input إلى المتغير myName

عندما يتم استدعاء التابع ()input فإن البرنامج ينتظر المستخدم إلى أن يقوم بإدخال نص معين ويضغط انتر.

في هذا البرنامج فإن المستخدم يقوم بإدخال اسمه (سلسلة نصية) والتي سوف يتم حفظها في المتغير myName

التابع ()input يعيد القيمة التي يقوم المستخدم بإدخالها

أثناء استدعاء التابع ()input لا يتم تمرير أي قيمة داخل القوسين () وهذا التابع يقوم فقط بإعادة القيمة التي يقوم المستخدم بإدخالها

```
hello.py
5. print('It is good to meet you, ' + myName)
```

في السطر الخامس قمنا باستدعاء التابع ()print مع استخدام معامل الجمع (+) من أجل دمج السلسلة النصية 'It is good to meet you' مع اسم المستخدم الذي يقوم بإدخاله.

تذكر أن جهاز الكمبيوتر يقوم فقط بتنفيذ الأوامر التي يطلبها البرنامج منه.

البرنامج السابق مبرمج لكي يطلب من المستخدم أن يقوم بإدخاله اسمه ويسمح له بإدخال قيمة نصية ثم يقول Hello world ويظهر السلسلة النصية التي قام المستخدم بإدخالها.

الكمبيوتر والبرنامج لا يهتمون بالقيمة النصية (الاسم) التي يقوم المستخدم بإدخالها إن كانت اسم المستخدم الحقيقي أو أي اسم آخر أو حتى أي كلمات أخرى ويتم عرض السلسلة النصية تماماً كما ادخلها المستخدم.

الخلاصة:

كتابة البرامج هي طريقة للتحدث مع الكمبيوتر، في هذا الفصل قمنا بجمع عدة تعليمات من لغة البايثون من أجل كتابة برنامج كامل يقوم بسؤال المستخدم عن اسمه ثم يرحب به

كل البرامج القادمة في هذا الكتاب سوف تكون أكثر تعقيد ولكن لا تقلق لأننا سنقوم بشرح هذه البرامج سطر سطر ويمكنك دائماً إدخال التعليمات في الشيل التفاعلية لترى العمل الذي تقوم به قبل استخدامها في البرنامج الكامل.

الآن سوف نبدأ ببرنامجنا الأول لفك تشفير الشيفرة العكسية reverse cipher



الشيفرة العكسية

محتوى هذا الفصل:

- التابع ()len
- حلقة while
- البيانات من النوع البولياني Boolean (المنطقي)
- معاملات المقارنة
- الأوامر الشرطية

“Every man is surrounded by a neighborhood of voluntary spies.”

Jane Austen

الشيفرة العكسية Reverse Cipher:

الشيفرة العكسية تقوم بتشفير الرسالة من خلال طباعة أحرف الرسالة بشكل عكسي.

مثلاً الرسالة "Hello world!" يتم تشفيرها بالشكل التالي "!dlrow olleH" من أجل فك التشفير يمكن عكس الرسالة المشفر من أجل الحصول على الرسالة الأصلية.

في هذه الشيفرة فإن خطوات التشفير وفك التشفير هي نفسها.

الشيفرة العكسية هي شيفرة ضعيفة جداً ويمكن كشفها من خلال النظر فقط إلى النص المشفر

سوف نستخدم هذه الشيفرة لأنها سهلة الشرح وسيكون من السهل كتابة برنامجنا الأول لفك التشفير لهذه الشيفرة.

الكود البرمجي لبرنامج الشيفرة العكسية:

في IDLE اختر file > New File وقم بكتابة الكود التالي ثم قم حفظه باسم reverseCipher.py واضغط F5 من أجل تنفيذ البرنامج

(لا تكتب أرقام الأسطر في الكود البرمجي)

Source code for reverseCipher.py

```
# Reverse Cipher
# http://inventwithpython.com/hacking (BSD Licensed)

message = 'Three can keep a secret, if two of them are dead.'
translated = ''

i = len(message) - 1
while i >= 0:
    translated = translated + message[i]
    i = i - 1

print(translated)
```

```
reverseCipher.py - C:/Python34/reverseCipher.py (3.4.2)
File  Edit  Format  Run  Options  Windows  Help

# Reverse Cipher
# http://inventwithpython.com/hacking (BSD Licensed)

message = '.daed era meht fo owt fi ,terces a peek nac eerhT'
translated = ''

i = len(message) - 1
while i >= 0:
    translated = translated + message[i]
    i = i - 1

print(translated)

Ln: 2 Col: 0
```

عند تنفيذ هذا البرنامج ستكون النتيجة كالتالي:

```
>>>
.daed era meht fo wot fi ,terces a peek nac eerhT
>>>
```

من أجل فك تشفير هذه الرسالة

'.daed era meht fo owt fi ,terces a peek nac eerhT'

قم بنسخ النص المشفر ولصقه في السطر الرابع ليتم حفظه في المتغير message

وتأكد من وجود علامات التنصيص في بداية ونهاية هذا النص

```
reverseCipher.py
4. message = '.daed era meht fo owt fi ,terces a peek nac eerhT'
```

الآن وعند تشغيل هذا البرنامج سنحصل على النص الأصلي بعد القيام بعملية فك التشفير

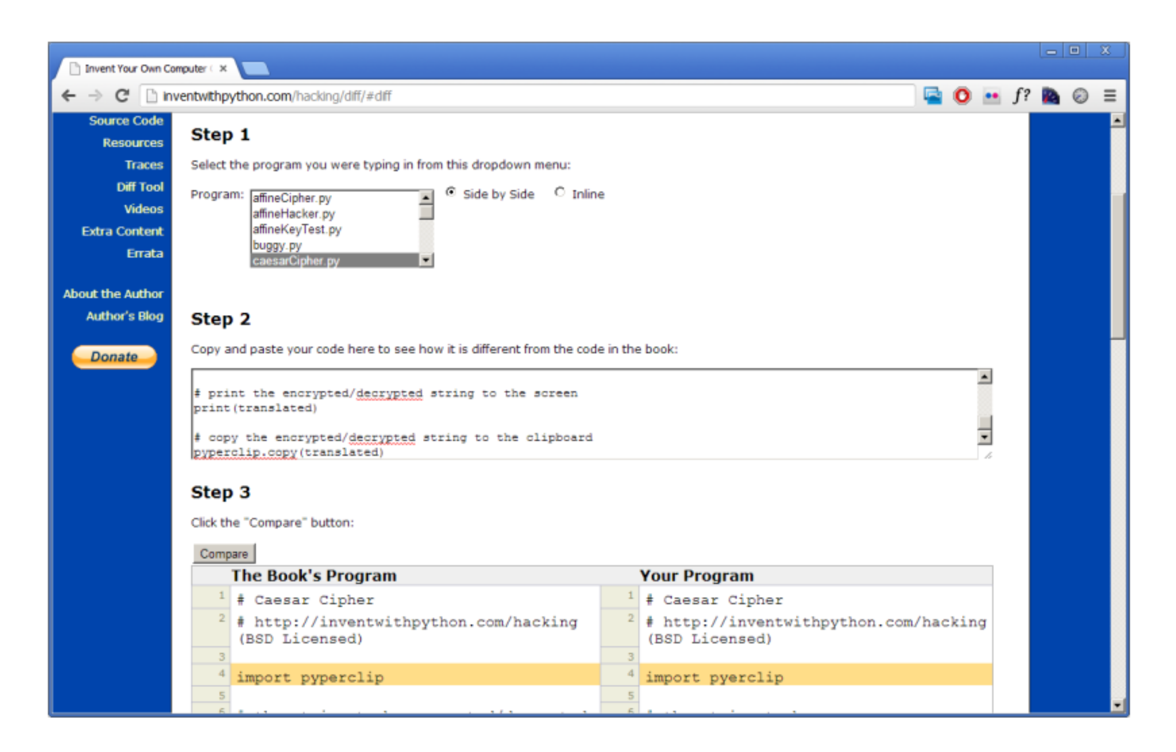
```
>>>
Three can keep a secret, if two of them are dead.
>>>
```

فحص الكود البرمجي بأداة Diff بشكل أون لاين:

رغم أنه يمكنك أن تقوم بنسخ ولصق الكود البرمجي أو تحميله من موقع الكتاب على الانترنت ولكن من المفيد أن تقوم بكتابة هذه البرامج بنفسك وهذا سوف يعطيك فكرة أفضل عن الكود في هذه البرامج ولكن في بعض الحالات سوف تقع بع فيض الأخطاء عندما تقوم بكتابة الكود البرمجي بنفسك.

من أجل مقارنة الكود المكتوب في هذا الكتاب مع الكود الذي قمت بكتابته بنفسك يمكنك ذلك من خلال نسخ الكود الذي قمت بكتابته ولصقه في الحقل النصي في المتصفح في الصفحة http://invpy.com/hackingdiff

ثم الضغط على زر Compare عندها سوف تقوم الأداة diff بإظهار أي اختلاف بين الكود الذي قمت أنت بكتابته والكود الأصلي الموجود في الكتاب وهي طريقة سهلة من أجل ايجاد الأخطاء الكتابية في الكود والتي تسبب ظهور رسائل خطأ



كيف يعمل هذا البرنامج:

reverseCipher.py

```
1. # Reverse Cipher
2. # http://inventwithpython.com/hacking (BSD Licensed)
```

أول سطرين في البرنامج هما عبارة عن تعليقات، السطر الأول يشرح المهمة التي يقوم بها البرنامج أما السطر الثاني فهو عنوان الموقع الذي يمكنك إيجاد هذا البرنامج فيه.

الرخصة DSD Licensed تعني أن هذا البرنامج هو مجاني ويمكن نسخه والتعديل عليه من قبل أي شخص طالما أن هذا البرنامج سوف ينسب إلى كاتبه الأصلي (في هذه الحالة النسبة إلى موقع الكتاب http://inventwithpython.comhacking)

نص رخصة BSD: Berkeley Software Distribution يمكن أن تجده في http://invpy.com/bsd

reverseCipher.py

```
4. message = 'Three can keep a secret, if two of them are dead.'
```

السطر الرابع يقوم بتخزين السلسلة النصية التي نريد تشفيرها في متغير باسم message

في كل مرة نريد فك تشفير أو تشفير أي سلسلة نصية جديدة فقط نقوم بكتابة هذه السلسلة بشكل مباشر في السطر الرابع في الكود البرمجي (في هذا البرنامج لم نستخدم التابع input() وبدل ذلك يجب على المستخدم كتابة الرسالة التي يريد تشفيرها أو فك تشفيرها بشكل مباشر في الكود البرمجي)

وفي كل مرة يجب تغير الكود البرمجي من أجل تشفير أو فك تشفير رسالة جديدة

reverseCipher.py

```
5. translated = ''
```

قمنا بتعريف متغير جديد باسم translated وهو يحوي حالياً على سلسلة نصية فارغة

(تذكر أن تعريف السلسلة النصية الفارغة يتم باستخدام زوج من علامات التنصيص المفردة وليس علامات التنصيص المزدوجة)

التابع ()len:

```
reverseCipher.py
7. i = len(message) - 1
```

السطر السادس هو سطر فارغ وبايثون سوف يتجاوزه، في السطر التالي من الكود قمنا بإسناد قيمة للمتغير i

القيمة المسندة هي العبارة الجبرية len(message) – 1

أول جزء من العبارة الجبرية هو len(message)

وهو استدعاء للتابع ()len، هذا التابع يقبل القيم النصية كمدخلات تمرر له أثناء عملية الاستدعاء ثم يقوم بإعادة قيمة عددية (عدد صحيح) تخبرنا بعدد الأحرف الموجودة في السلسلة النصية الممررة لهذا التابع (طول السلسلة النصية).

في البرنامج السابق قمنا بتمرير المتغير message إلى التابع ()len

Len(massage) سوف يخبرنا بعدد أحرف السلسلة النصية المخزنة في المتغير message

جرب المثال التالي:

```
>>> len('Hello')
5
>>> len('')
0
>>> spam = 'Al'
>>> len(spam)
2
>>> len('Hello' + ' ' + 'world!')
12
>>>
```

من القيمة التي يعيدها التابع ()len نعرف أن السلسلة النصية 'Hello' تملك 5 أحرف

وأن السلسلة النصية الفارغة تملك 0 حرف

وإذا قمت بتخزين السلسلة النصية 'Al' في متغير ومن ثم قمنا باستدعاء التابع ()len ومررنا سم المتغير لهذا التابع فسوف يقوم التابع بإعادة القيمة 2

إذا مررنا العبارة 'Hello' + ' ' + 'world!' إلى التابع ()len فسوف يعيد القيمة 12

لأن المسافة الفارغة وعلامة التعجب يتم اعتبارهم كأحرف.

السطر السابع في البرنامج يقوم بإيجاد عدد أحرف السلسة النصية المخزنة في المتغير message ويطرح منها العدد واحد ثم يقوم بتخزين العدد الناتج في المتغير i

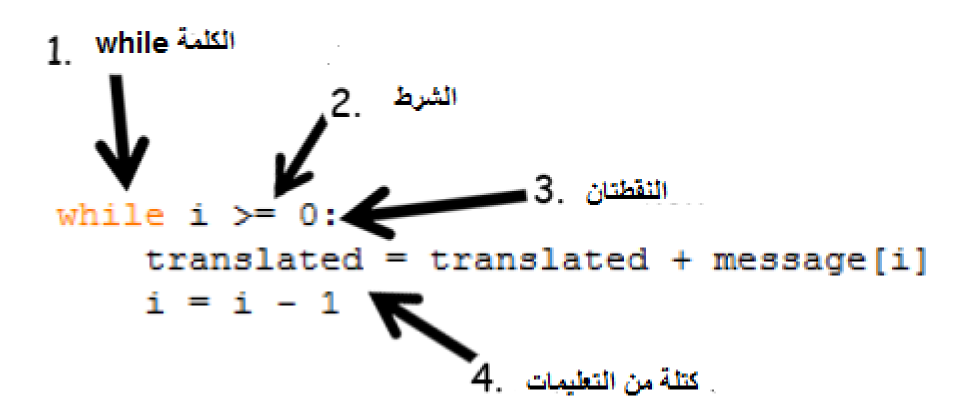
العدد المخزن في المتغير i يستخدم كفهرسة من أجل الإشارة للحرف الأخير في السلسلة النصية المخزنة في المتغير message.

الحلقة while:

```
reverseCipher.py
8. while i >= 0:
```

الحلقة while مكونة من أربعة أجزاء:

١. الكلمة while
٢. عبارة جبرية وتسمى الشرط والتي تحسب قيمة منطقية (قيمة بوليانية) True or False (سيتم شرحها في الفصل القادم)
٣. النقطتان :
٤. كتلة من التعليمات (كما في السطرين 9 and 10)



لتتمكن من فهم الحلقة while يجب أن تفهم المتحولات المنطقية (البوليانية) وعبارات المقارنة

البيانات من النوع المنطقي (البولياني):

هذا النوع من البيانات يحوي فقط على قيمتين صح أو خطأ True or False

وهي حساسة لحالة الأحرف، يجب أن تبدأ دائماً بحرف كبير T and F وباقي الأحرف صغيرة

وهي ليست قيم نصية ولا يجب أن تضعها بين علامات التنصيص وهي تستخدم مع معاملات المقارنة (سوف تشرح لاحقاً في هذا الفصل)

مثل أي نوع من البيانات الأخرى فإن القيم البوليانية يمكن تخزينها ضمن المتغيرات

جرب المثال التالي:

```
>>> spam = True
>>> spam
True
>>> spam = False
>>> spam
False
>>>
```

معاملات المقارنة:

في السطر الثامن من البرنامج، انظر إلى العبارة بعد الكلمة while

```
reverseCipher.py
8. while i >= 0:
```

i >= 0 تحوي على قيمتين (القيمة الموجودة في المتغير i والقيمة 0) ومعامل المقارنة >= والذي يلفظ أكبر أو يساوي.

معاملات المقارنة تظهر بالجدول التالي:

الإشارة	العلمية
<	أصغر من
>	أكبر من
<=	أصغر أو يساوي
>=	أكبر أو يساوي
==	يساوي
!=	لا يساوي

ادخل العبارات التالية في الشيل التفاعلية وشاهد النتيجة المنطقية (البوليانية) التي يتم حسابها

```
>>> 0 < 6
True
>>> 6 < 0
False
>>> 50  < 10.5
False
>>> 10.5 < 11.3
True
>>> 10 < 10
False
>>>
```

العبارة 6 > 0 تعيد القيمة المنطقية (البوليانية) True لأن العدد 0 هو أصغر من العدد 6

العبارة 0 > 6 تعيد القيمة المنطقية False لأن العدد 6 ليس أصغر من العدد 0

العبارة 10.5 > 50 تعيد القيمة المنطقية False لأن العدد 50 ليس أصغر من العدد 10.5

العبارة 11.3 > 10.5 تعيد القيمة True لأن العدد 10.5 هو أصغر من العدد 11.3

العبارة 10 > 10 تعيد القيمة False لأن العدد 10 ليس أصغر من نفسه

جرب العبارات الشرطية التالي:

```
>>> 10 <= 20
True
>>> 10 <= 10
True
>>> 10 >= 20
False
>>> 20 >= 20
True
>>>
```

```
>>> 10 == 10
True
>>> 10 == 11
False
>>> 10 != 10
False
>>> 10 != 11
True
>>> 'Hello' == 'Hello'
True
>>> 'Hello' == 'Good bye'
False
```

```
>>> 'Hello' == 'HELLO'
False
>>> 'Good bye' != 'Hello'
True
>>>
```

انتبه للفرق بين = و ==

إشارة = تستخدم لإسناد قيمة معينة لمتغير أما == تستخدم للسؤال عن تساوي قيمتين

إذا كنت تريد سؤال بايثون عن تساوي قيمتين استخدم ==

إذا كنت تريد من البايثون أن يقوم بإسناد قيمة إلى متغير استخدم =

الأعداد والسلاسل النصية هي دائما قيم غير متساوية، جرب المثال التالي:

```
>>> 42 == 'Hello'
False
>>> 42 == '42'
False
>>> 10 == 10.0
True
>>>
```

لاحظ أن '42' أو أي قيمة رقيميه أخرى عندما يتم كتابتها داخل علامات التنصيص تصبح سلسلة نصية.

الشروط:

الشرط هو العبارة التي تكتب بعد تعليمة while or if

(التعليمة if سوف نتعرف عليها لاحقاً في فصل قادم)

الشروط عادتاً تحوي على معاملات المقارنة ولكن الشروط تبقى عبارات جبرية.

كتلة التعليمات:

كتلة التعليمات هي عبارة عن سطر أو أكثر من الكود البرمجي تكون مجموعة مع بعضها بنفس الفراغ (الهامش) في بداية كل سطر

يمكنك معرفة بداية الكتلة ونهايتها من خلال النظر إلى المسافة في بداية الأسطر

الكتلة تبدأ عندما يكون السطر قد ازيح بمقدار أربع مسافات عن الحافة اليسرى وكل الأسطر اللاحقة التي تكون مزاحة بمقدار أربع مسافات هي جزء من الكتلة.

عندما يكون السطر مزاح بمقدار أربع مسافات أخرى (مزاح بمقدار ثمانية عن البداية) فهذا يعني أن كتلة جديدة من التعليمات قد بدأت داخل الكتلة الأولى

```
1. codecodecodecodecodecodecode # لا يحوي على مسافة بدائية
2. ▪▪▪▪codecodecodecodecodecodecodecodecode # يحوي على مسافة بمقدار أربع فراغات
3. ▪▪▪▪codecodecodecodecodecodecode # يحوي على مسافة بمقدار أربع فراغات
4. ▪▪▪▪▪▪▪▪codecodecodecodecodecodecodecodecode # يحوي على مسافة بمقدار ثمانية
فراغات
5. ▪▪▪▪codecodecodecodecode # يحوي على مسافة بمقدار أربع فراغات
6.
7. ▪▪▪▪codecodecodecodecodecode # يحوي على مسافة بمقدار أربع فراغات
8. codecodecodecodecodecodecodecodecodecodecode# لا يحوي على مسافة بدائية
```

- السطر الأول لا يحوي على أي مسافة بدائية
- السطر الثاني يحوي على مسافة بدائية مقدارها أربع فراغات وهذا يعني أن كتلة جديدة قد بدأت.
- السطر الثالث يحوي أيضاً على مسافة بدائية مقدارها أربعة فراغات وهذا يعني أن الكتلة السابقة مستمرة.
- السطر الرابع يحوي على مسافة بدائية مقدارها ثمانية فراغات وهذا يعني أن كتلة ثانية قد بدأت داخل الكتلة الأولى.
- السطر الخامس يحوي على مسافة بدائية مقدارها أربعة فراغات وهذا يعني أن الكتلة الثانية قد انتهت وأن الكتلة الأولى مستمرة.
- السطر السادس هو فارغ وسوف يتجاهله البايثون.

- السطر السابع يحوي على مسافة بدائية مقدارها أربعة فراغات وهذا يعني أنه استمرار للكتلة التي بدأت في السطر الثاني.
- السطر الثامن لا يحوي على مسافة بدائية وهذا يعني ان الكتلة التي بدأت في السطر الثاني قد انتهت في السطر السابع.

(قيمة المسافة أربعة فراغات ليست قيمة إجبارية ويمكن أن تكون أي قيمة أخرى ولكن جرت العادة أن تكون هذه المسافة عبارة عن أربعة فراغات)

تعليمات الحلقة while:

reverseCipher.py

```
8. while i >= 0:
9.     translated = translated + message[i]
10.    i = i - 1
11.
12. print(translated)
```

التعليمة while في السطر الثامن تخبر البياثون أن يقوم أولاً بفحص الشرط (i >= 0) إذا كانت نتيجة هذا الشرط هي True (الشرط محقق) فإن البرنامج سوف يقوم بتنفيذ التعليمات الموجودة داخل الكتلة التي تلي تعليمة while (الكتلة هي السطرين 9 and 10)

أما إذا كتنت نتيجة تنفيذ الشرط هي False (الشرط غير محقق) فإن البرنامج سوف يتجاوز الكود الموجود في الكتلة التالية ويقفز لتنفيذ أول سطر من الكود الذي يلي الكتلة (السطر 12)

إذا كان الشرط محقق فإن البرنامج سوف يبدأ بتنفيذ التعليمات الموجودة في الكتلة وعندما ينتهي من تنفيذ أخر تعليمة في هذه الكتلة فسوف يعود إلى حلقة while مرة ثانية ليرى نتيجة الشرط مرة ثانية وإذا بقي الشرط محقق فإن البرنامج سوف يقوم بتنفيذ التعليمات الموجودة داخل الكتلة مرة ثانية ثم يعود إلى حلقة while ويفحص حالة الشرط مرة أخرى فإذا أصبح الشرط غير محقق عندها فإن البرنامج سوف يتجاوز أسطر الكتلة ويقفز لتنفيذ أول سطر بعد الكتلة.

العبارة while i >= 0 تعني: طالما أن قيمة المتغير i هي أكبر أو تساوي الصفر، استمر في تنفيذ التعليمات الموجودة في الكتلة التالية.

ازدياد السلسلة:

تذكر أنه في السطر السابع قد تم تعين القيمة الأولية للمتغير i وهي طول السلسلة النصية المخزنة في المتغير message منقوص منها واحد وأن الحلقة while في السطر الثامن تستمر في تنفيذ التعليمات الموجودة داخل الكتلة التي تليها إلى أن يصبح الشرط i >= 0 غير محقق False

reverseCipher.py

```
7. i = len(message) - 1
8. while i >= 0:
9.     translated = translated + message[i]
10.    i = i - 1
11.
12. print(translated)
```

كتلة التعليمات التي تلي while تحوي على السطرين 9 and 10

السطر 9 يقوم بإسناد قيمة جديدة للمتغير translated ، هذه القيمة تزداد في كل مرة بالحرف ذو قيمة الفهرسة i من السلسلة النصية المخزنة في المتغير message وبهذه الطريقة فإن قيمة السلسلة المخزنة في المتغير translated تزداد إلى أن تصبح السلسلة النصية الكاملة الغير مشفرة.

في السطر العاشر يتم اسناد قيمة جديدة للمتغير i وهذه القيمة هي القيمة الحالية لهذا المتغير منقوصاً منها واحد.

السطر 12 لا يحوي على مسافة بدائية وهذا يخبر بايثون بأن كتلة التعليمات التي تلي while قد انتهت وبدلاً من أن يقوم البرنامج بتنفيذ السطر 12 فهو يعود إلى السطر 8 ليقوم بفحص حالة الشرط مرة ثانية، إذا كان الشرط محقق فسوف يقوم بتنفيذ التعليمات الموجودة داخل الكتلة مرة ثانية (السطرين 9 and 10) ويستمر بتنفيذهما إلى أن يصبح الشرط غير محقق False (عندما تصبح قيمة i أصغر من الصفر) وفي هذه الحالة فإن البرنامج سوف يقوم بتنفيذ أول سطر بعد الكتلة وهو السطر 12

المتغير i يبدأ بقيمة الفهرسة لآخر حرف في السلسلة النصية المخزنة في المتغير message.

المتغير translated يبدأ بقيمة سلسلة فارغة ثم (في داخل الحلقة) يأخذ هذا المتغير قيمة message[i] (والتي هي الحرف الأخير من السلسلة النصية الموجودة في المتغير message) حيث أن المتغير i يحوي على قيمة الفهرسة الخاصة بآخر حرف في السلسلة.

ثم يتم أنقاص قيمة المتغير i بمقدار واحد وهذا يعني أن message[i] أصبحت تشير إلى الحرف ما قبل الأخير في السلسلة النصية ويتم إضافته إلى قيمة المتغير translated.

قيمة المتغير i تستمر بالتناقص إلى أن يصبح message[i] هو أول حرف في السلسلة ويتم إضافته كآخر حرف في السلسلة المخزنة في المتغير translated.

وهذا يجعل القيمة المخزنة في المتغير translated هي عكس القيمة الموجودة في المتغير message وعندها تصبح قيمة المتغير i = -1 وعندها يصبح الشرط غير محقق وسوف يقفز البرامج لتنفيذ السطر 12

reverseCipher.py

```
12. print(translated)
```

في نهاية البرنامج (السطر 12) قمنا بطباعة محتوى المتغير translated وهي السلسلة النصية ('.daed era meht fo owt fi ,terces a peek nac eerhT')

إذا لم تفهم كيف يقوم هذا البرنامج بعكس السلسلة النصية حاول أن تقوم بإضافة سطر جديد داخل الحلقة while وهو:

```
print(i, message[i], translated)
```

ليصبح الكود البرمجي كما في الشكل التالي:

reverseCipher.py

```
 8. while i >= 0:
 9.     translated = translated + message[i]
10.     print(i, message[i], translated)
11.     i = i - 1
12.
13. print(translated)
```

التابع ()print سوف يقوم بطباعة ثلاث قيم:

١. قيمة المتغير i
٢. [message[i
٣. قيمة المتغير translated

وذلك في كل مرة يتم فيها تنفيذ التعليمات داخل الكتلة في الحلقة while

هذا سوف يشرح خطوات تكرار الحلقة

الآن يمكنك تشغيل البرنامج و مشاهدة كيف تتغير قيمة المتغير translated

```
>>>
48 . .
47 d .d
46 a .da
45 e .dae
44 d .daed
43   .daed
42 e .daed e
41 r .daed er
40 a .daed era
39   .daed era
38 m .daed era m
37 e .daed era me
36 h .daed era meh
35 t .daed era meht
34   .daed era meht
33 f .daed era meht f
32 o .daed era meht fo
31   .daed era meht fo
30 o .daed era meht fo o
29 w .daed era meht fo ow
28 t .daed era meht fo owt
27   .daed era meht fo owt
26 f .daed era meht fo owt f
25 i .daed era meht fo owt fi
24   .daed era meht fo owt fi
23 , .daed era meht fo owt fi ,
22 t .daed era meht fo owt fi ,t
21 e .daed era meht fo owt fi ,te
20 r .daed era meht fo owt fi ,ter
19 c .daed era meht fo owt fi ,terc
18 e .daed era meht fo owt fi ,terce
17 s .daed era meht fo owt fi ,terces
16   .daed era meht fo owt fi ,terces
15 a .daed era meht fo owt fi ,terces a
```

```
14   .daed era meht fo owt fi ,terces a
13 p .daed era meht fo owt fi ,terces a p
12 e .daed era meht fo owt fi ,terces a pe
11 e .daed era meht fo owt fi ,terces a pee
10 k .daed era meht fo owt fi ,terces a peek
9   .daed era meht fo owt fi ,terces a peek
8 n .daed era meht fo owt fi ,terces a peek n
7 a .daed era meht fo owt fi ,terces a peek na
6 c .daed era meht fo owt fi ,terces a peek nac
5   .daed era meht fo owt fi ,terces a peek nac
4 e .daed era meht fo owt fi ,terces a peek nac e
3 e .daed era meht fo owt fi ,terces a peek nac ee
2 r .daed era meht fo owt fi ,terces a peek nac eer
1 h .daed era meht fo owt fi ,terces a peek nac eerh
0 T .daed era meht fo owt fi ,terces a peek nac eerhT
.daed era meht fo owt fi ,terces a peek nac eerhT
>>>
```

السطر الأول الذي يحوي على " . . 48 " يظهر قيمة i, message[i] and translated

حيث أن:

i = 48

message[i] = . (وهي أخر حرف في السلسلة النصية)

translated = .

بعدها يتم إنقاص قيمة المتغير i بقدار واحد ليصبح:

i = 47

message[i] = d

translated = .d

وتستمر هذه العملية إلى أن يصبح

i = 0

message[i] = T

translated = .daed era meht fo owt fi ,terces a peek nac eerhT

شرح البرنامج خطوة بخطوة:

سوف نقوم بشرح البرنامج خطوة بخطوة بنفس الطريق التي يقوم بها مترجم البايثون بالتنفيذ.

مترجم البايثون يبدأ بتنفيذ البرنامج من الأعلى من أول سطر ثم ينتقل إلى الأسفل ليقوم بتنفيذ التعليمات التالية.

الأسطر الفارغة والتعليقات سوف يتم تجاهلها.

الجدول التالي يظهر شرح لكل سطر بنفس الطريقة التي يقوم مترجم بايثوزن بتنفيذ هذا البرنامج:

reverseCipher.py

```
1. # Reverse Cipher
2. # http://inventwithpython.com/hacking (BSD Licensed)
3.
4. message = 'Three can keep a secret, if two of them are dead.'
5. translated = ''
6.
7. i = len(message) - 1
8. while i >= 0:
9.     translated = translated + message[i]
10.    i = i - 1
11.
12. print(translated)
```

الخطوة ١	السطر ١	تعليق وسيتم تجاهله
الخطوة ٢	السطر ٢	تعليق وسيتم تجاهله
الخطوة ٣	السطر ٣	تخزين السلسلة النصية 'Three can keep a secret, if two of them are

		dead.' في المتغيرmessage
الخطوة ٤	السطر ٥	تخزين السلسلة النصية الفارغة '' في المتغير translated
الخطوة ٥	السطر ٧	len(message) -1 = 48 يتم تخزين القيمة 48 في المتغير i
الخطوة ٦	السطر ٨	شرط الحلقة while (i >= 0) محقق والبرنامج سوف يقوم بتنفيذ التعليمات داخل كتلة التعليمات التالية
الخطوة ٧	السطر ٩	يتمم حساب translated + message[i] والتي تساوي القيمة النصية ' . ' ويتم تخزين هذه القيمة في المتغير translated
الخطوة ٨	السطر ١٠	يتم حساب i – 1 والتي تساوي 47 ويتم تخزين هذه القيمة في المتغير i
الخطوة ٩	السطر ٨	عندما ينتهي البرنامج من تنفيذ آخر تعليمة في الكتلة يعود إلى بداية الحلقة ليقوم بإعادة فحص حالة الشرط مرة ثانية والشرط محقق للمرة ثانية والبرنامج سوف يقوم بتنفيذ التعليمات داخل الكتلة للمرة ثانية
الخطوة ١٠	السطر ٩	يتم حساب translated + message[i] والتي تساوي 'd. ' ويتم تخزين هذه السلسلة النصية في المتغير translated
الخطوة ١١	السطر ١٠	يتم حساب i – 1 والتي تساوي 46 وسيتم حفظ هذه القيمة في المتغير i
الخطوة ١٢	السطر ٨	يتم العودة لفحص حالة الشرط وهو مازال محقق وينتقل البرنامج لتنفيذ التعليمات داخل الكتلة مرة أخرى
الخطوة ١٣ إلى الخطوة ١٤٩		يستمر تنفيذ التعليمات داخل الكتلة إلى أن تصبح قيمة المتغير i = 0 وقيمة المتغير translated = ' .daed era meht fo owt fi ,terces

		a peek nac eerh'
الخطوة ١٥٠	السطر ٨	يتم إعادة فحص الشرط وهو محقق 0 =< 0
الخطوة ١٥١	السطر ٩	يتم حساب translated + message[i] والتي تساوي 'daed era meht foowt fi ,terces a peek nac eerhT' ويتك حفظ هذه السلسة النصية في المتغير translated
الخطوة ١٥٢	السطر ١٠	يتم حساب i – 1 والتي تساوي 1- ويتم تخزين هذه القيمة في المتغير i
الخطوة ١٥٣	السطر ٨	الشرط 0 =< 1- غير محقق والبرنامج سوف يتجاوز التعليمات داخل الكتلة وسوف ينتقل لتنفيذ السطر 12
الخطوة ١٥٤	السطر ١٢	تصبح قيمة المتغير translated هي السلسة النصية '.daed era meht fo owt fi , terces a peek nac eerhT' ويتم تمرير استدعاء التابع ()print وتمرير هذه القيمة له من أجل أن يقوم بطباعتها على الشاشة
		لا يوجد أي سطر بعد السطر 12 لذلك يتوقف البرنامج

استخدم التابع input() في برنامجنا:

البرامج في هذا الكتاب مصممة لتقوم بتشفير أو فك تشفير الرسالة أو السلسلة النصية التي يتم كتابتها بشكل مباشر في الكود البرمجي، يمكن تعديل هذه الخاصية من خلال استدعاء التابع input() والذي يسمح للمستخدم بإدخال السلسلة النصية التي يرغب بفك تشفيرها، هذا التابع يظهر المؤشر للمستخدم ويتوقف عن تنفيذ البرنامج إلى أن يقوم المستخدم بإدخال القيمة النصية والضغط على انتر.

يمكنك تغير السطر الرابع في البرنامج ليصبح كالتالي:

```
reverseCipher.py
4. message = input('Enter message: ')
```

الآن وعند تنفيذ البرنامج سوف يظهر المؤشر prompt على الشاشة وينتظر إلى أن يقوم المستخدم بإدخال الرسالة والضغط على انتر

الرسالة التي يقوم المستخدم بإدخالها يجب أن تكون سلسلة نصية وسيتم حفظها في المتغير message

```
Enter message: Hello world!
!dlrow olleH
>>>
```

الخلاصة:

الآن وبعد أن تعلمنا كيفية التعامل مع النصوص يمكننا البدء بكتابة البرامج التي يمكن أن تتفاعل مع المستخدم.

السلاسل النصية هي نوع من البيانات والذي سوف نستخدمه في برامجنا ويمكننا استخدام المعامل ' + ' من أجل جمع السلاسل النصية مع بعضها البعض، كما يمكننا استخدام الفهرسة والاقتطاع من أجل خلق سلسلة جديدة كجزء من سلسلة نصية أخرى.

التابع ()len: يتم تمرير سلسلة نصية له ويقوم بإعادة رقم صحيح يدل على عدد الأحرف في هذه السلسة النصية.

نوع البيانات المنطقي (البولياني) له فقط قيمتين: True and False.

معاملات المقارنة =< , => , < , > , =! , == تستخدم لمقارنة قيمتين وتظهر الناتج على شكل قيمة منطقية (بوليانية).

الشروط هي عبارات جبرية تستخدم في عدة أنواع مختلفة من التعليمات.

الحلقة while تستمر بتنفيذ التعليمات داخل الكتلة التي تليها طالما أن الشرط محقق True.

الكتلة هي مجموعة من الأسطر التي لها نفس الإزاحة (المسافة البدائية) ويمكن أن تحوي في داخلها على كتل جديدة.

الأجراء المتبع في البرنامج هو البداية بمتغير يحوي على سلسلة نصية فارغة ومن ثم زيادة هذه السلسلة حرف بحرف إلى أن نحصل على السلسلة المطلوبة.



شيفرة قيصر

محتوى هذا الفصل:

- التعليمة import
- الثوابت constants
- الطريقة method: upper()
- حلقة for
- تعليمات if, elif and else
- المعاملات in and not in
- الطريقة fine()

“BIG BROTHER IS WATCHING YOU.”

“1984” by George Orwell

تجهيز البرنامج:

في الفصل الأول تعلمنا عدة طرق من أجل القيام بتشفير قيصر، الآن سوف نستخدم برنامج من أجل القيام بهذه المهمة.

الشيفرة العكسية دائماً تقوم بالتشفير بنفس الطريقة ولكن تشفير قيصر يستخدم المفاتيح keys حيث يتم تشفير الرسالة بطرق مختلفة بالاعتماد على قيمة المفتاح المستخدم.

المفتاح في شيفرة قيصر هو عدد صحيح ضمن المجال 0 to 25.

إذا عرف محلل الشيفرة cryptanalyst أن التشفير المستخدم هو تشفير قيصر فهذا غير كافي ويجب أن يعرف قيمة المفتاح المستخدم في عملية التشفير من أجل كسر أو فك هذه الشيفرة.

الكود البرمجي لشيفرة قيصر:

قم بكتابة الكود التالي في محرر النصوص واحفظه باسم caesarCipher.py واضغط F5 من أجل تنفيذ البرنامج

يجب أن تقوم أولاً بتحميل الوحدة pyperclip.py module ووضع هذا الملف في نفس المجلد الموجود فيه الملف caesarCipher.py

يمكنك تحميل هذا الموديول من http://invpy.com/pypreclip.py (عملية التحميل هي عبارة عن نسخ الكود الموجود في هذه صفحة الويب السابقة ولصقه وحفظه في محرر النصوص باسم pyperclip.py)

Source code for caesarCipher.py

```
# Caesar Cipher
# http://inventwithpython.com/hacking (BSD Licensed)

import pyperclip

# the string to be encrypted/decrypted
message = 'This is my secret message.'

# the encryption/decryption key
key = 13

# tells the program to encrypt or decrypt
mode = 'encrypt' # set to 'encrypt' or 'decrypt'

# every possible symbol that can be encrypted
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

# stores the encrypted/decrypted form of the message
translated = ''

# capitalize the string in message
message = message.upper()

# run the encryption/decryption code on each symbol in the message string
for symbol in message:
    if symbol in LETTERS:
        # get the encrypted (or decrypted) number for this symbol
        num = LETTERS.find(symbol) # get the number of the symbol
        if mode == 'encrypt':
            num = num + key
        elif mode == 'decrypt':
            num = num - key

        # handle the wrap-around if num is larger than the length of
        # LETTERS or less than 0
        if num >= len(LETTERS):
            num = num - len(LETTERS)
        elif num < 0:
            num = num + len(LETTERS)

        # add encrypted/decrypted number's symbol at the end of translated
        translated = translated + LETTERS[num]

    else:
        # just add the symbol without encrypting/decrypting
        translated = translated + symbol

# print the encrypted/decrypted string to the screen
print(translated)

# copy the encrypted/decrypted string to the clipboard
pyperclip.copy(translated)
```

عند تشغيل هذا البرنامج سوف تظهر النتيجة التالية:

```
>>>
GUVF VF ZL FRPERG ZRFFNTR.
>>>
```

تم تشفير السلسلة النصية 'This is my secret message.' بشيفرة قيصر باستخدام المفتاح 13

هذا البرنامج عندما يعمل يقوم بشكل أتوماتيكي بنسخ النص المشفر ويمكنك لصقه في ملف نصي أو في رسالة ايميل.

بهذه الطريقة يمكنك بسهولة أخذ النص المشفر إلى خارج البرنامج وإرسالها إلى شخص أخر من أجل فك التشفير قم بلصق النص المشفر في السطر 7 ليتم اسنادها إلى المتغير message ثم قم بتغيير السطر 13 ليصبح mode = 'decrypt'

caesarCipher.py

```
 6. # the string to be encrypted/decrypted
 7. message = 'GUVF VF ZL FRPERG ZRFFNTR.'
 8.
 9. # the encryption/decryption key
10. key = 13
11.
12. # tells the program to encrypt or decrypt
13. mode = 'decrypt' # set to 'encrypt' or 'decrypt'
```

الآن وعند تنفيذ هذا البرنامج سوف نحصل على النتيجة التالية:

```
>>>
THIS IS MY SECRET MESSAGE.
>>>
```

إذا حصلت على رسالة الخطأ التالية:

```
Traceback (most recent call last):
  File "C:/Python34/caesarCipherde.py", line 4, in <module>
    import pyperclip
ImportError: No module named 'pyperclip'
>>>
```

هذا يعني أنك لم تقم بوضع الملف pyperclip.py في المجلد الصحيح (نفس المجلد الذي قمت بحفظ البرنامج بداخله)

إذا استمر هذا الخطأ ولم تتمكن من إصلاحه فقط قم بحذف السطرين 4 and 52 من الكود

هذا سوف يخلص الكود من التعليمات التي تعتمد على pyperclip module

الغاية من pyperclip module هو نسخ النص المشفر بشكل أتوماتيكي بعد تنفيذ البرنامج.

كيف يعمل هذا البرنامج:

سنقوم بشرح كل سطر من كود هذا البرنامج

استيراد الوحدات باستخدام التعليمة import:

caesarCipher.py

```
1. # Caesar Cipher
2. # http://inventwithpython.com/hacking (BSD Licensed)
3.
4. import pyperclip
```

في السطر الرابع استخدمنا التعليمة import

البايثون يحوي على العديد من التوابع المبنية في داخله، بعض التوابع موجودة في برامج منفصلة تسمى الموديول (الوحدة) Module وهي عبارة عن برنامج بايثون يحوي على توابع إضافية والتي يمكنك أن تستخدمها في برامجك.

في هذا البرنامج قمنا باستيراد الوحدة (الموديول) pyperclip من أجل أن نستطيع استدعاء التابع pyperclip.copy()

تعليمة الاستيراد import مكونة من الكلمة import وتليها اسم الوحدة المراد استيرادها.

في السطر الرابع قمنا باستيراد pyperclip module والذي يحوي على عدد من التوابع المتعلقة بنسخ ولصق النصوص من وإلى الذاكرة

caesarCipher.py

```
# the string to be encrypted/decrypted
message = 'This is my secret message.'

# the encryption/decryption key
key = 13

# tells the program to encrypt or decrypt
mode = 'encrypt' # set to 'encrypt' or 'decrypt'
```

في السطور السابقة قمنا بإعداد ثلاث متغيرات:

المتغير message يحوي على السلسلة النصية التي سيتم تشفيرها أو فك تشفيرها.

المتغير key يحوي على عدد صحيح وهو مفتاح التشفير encryption key

المتغير mode يحوي على إحدى القيمتين 'encrypt' (والتي تخبر البرنامج بالقيام بتشفير الرسالة) أو 'decrypt' (والتي تخبر البرنامج بالقيام بفك تشفير الرسالة)

الثوابت:

caesarCipher.py

```
# every possible symbol that can be encrypted
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

نحن بحاجة أيضاً إلى سلسلة نصية تحوي على كل الأحرف الأبجدية الكبيرة وبشكل مرتب. قمنا بإسناد السلسة النصية التي تحوي على كل الأحرف الأبجدية الكبيرة إلى المتغير LETTERS وهذه السلسلة تحوي على كل الأحرف التي يمكن أن يتم تشفيرها في برنامجنا هذه المجموعة من الأحرف تسمى مجموعة الرموز symbol set

في نهاية هذا الفصل سوف تتعلم كيف تقوم بتوسيع هذه المجوعة لتحوي على رموز بالإضافة للحروف.

الطريقة ()upper :

```
                                                        caesarCipher.py
18. # stores the encrypted/decrypted form of the message
19. translated = ''
20.
21. # capitalize the string in message
22. message = message.upper()
```

في السطر 19 يم حفظ سلسلة نصية فارغة في المتغير translated بشكل مشابه لبرنامج الشيفرة العكسية، وعند نهاية تنفيذ البرنامج فإن المتغير translated سوف يحوي على كامل السلسلة النصية المشفرة أو الغير مشفرة ولكن الآن يبدأ بسلسلة نصية فارغة.

في السطر 22 قمنا بإسناد قيمة للمتغير message ، هذه القيمة هي ناتج تطبيق الطريقة ()upper على القيمة المحفوظة في المتغير message.

الطريقة method هي مثل التابع function (باستثناء أنها تلحق بقيم غير محتواه في الوحدات non-module value كما في السطر 22 حيث أن المتغير يحوي على قيمة)

اسم الطريقة هو ()upper ويتم استدعائها وتطبيقها على القيمة النصية المحفوظة في المتغير message

التابع function ليس طريقة method لأنه موجود في الموديول module.

كما سوف نرى في السطر 52 الذي يتم فيه استدعاء ()pyperclio.copy ولكن pyperclip هي وحدة module والتي تم استيرادها في السطر الرابع، لذلك فإن ()copy هو ليس طريقة method وهو تابع لأنه موجود داخل pyperclip module

إذا كان هذا الأمر مربك لك فيمكنك دائماً تسمية الطرق والتوابع باسم تابع.

معظم أنواع البيانات (مثل السلاسل النصية) لها طرق methods.

السلاسل النصية تملك طرق ()upper() and lower والتي تقوم بتحويل أحرف السلسلة النصية إلى أحرف صغيرة أو كبيرة.

جرب المثال التالي:

```
>>> 'Hello world!'.upper()
'HELLO WORLD!'
>>> 'Hello world!'.lower()
'hello world!'
>>>
```

```
>>> fizz = 'Hello world!'
>>> fizz.upper()
'HELLO WORLD!'
>>> fizz
'Hello world!'
>>>
```

حلقة for:

caesarCipher.py

```
# run the encryption/decryption code on each symbol in the message string
for symbol in message:
```

حلقة for تستخدم من أجل تكرار أو الدوران عبر قائمة أو سلسلة نصية وبشكل مختلف عن حلقة while التي تحوي على شرط فإن حلقة for تحوي على 6 أجزاء وهي:

١. الكلمة for
٢. اسم المتغير
٣. الكلمة in
٤. القيمة النصية أو المتغير الذي يحوي على قيمة نصية
٥. النقطتان :
٦. كتلة من التعليمات



في كل مرة يتم فيها تنفيذ البرنامج فإنه سوف يقوم بتكرار تنفيذ كتلة التعليمات داخل الحلقة.

جرب المثال التالي في الشيل التفاعلية

```
>>> for letter in 'Syria':
        print(letter)


S
y
r
i
a
>>>
```

حلقة while المكافئة لحلقة for:

حلقة for تشبه حلقة while

يمكنك كتابة حلقة while تقوم بنفس العمل الذي تقوم به الحلقة for من خلال إضافة بعض التعليمات كما في المثال التالي:

```
>>> i = 0
>>> while i < len('Syria'):
        letter = 'Syria'[i]
        print('The letter is ' + letter)
        i = i + 1


The letter is S
The letter is y
The letter is r
The letter is i
The letter is a
>>>
```

لاحظ أن حلقة while السابقة تقوم بنفس العمل الذي تقوم به حلقة for ولكن حلقة for هي أبسط وأسهل

قبل أن تستطيع فهم الأسطر 26 to 32 من برنامج شيفرة قيصر يجب أن تتعلم أولاً التعليمات if, elif and eles والمعاملات in and not in والطريقة find()

التعليمة if:

التعليمة if تقرأ (إذا كان الشرط محقق True قم بتنفيذ كتلة التعليمات التالية وإذا لم يكن الشرط محقق False تجاوز كتلة التعليمات)

افتح محرر النصوص واكتب البرنامج التالي ثم قم بحفظه باسم password.py واضغط F5 من أجل تنفيذه

Source code for password.py

```
print('What is the password?')
password = input()
if password == 'rosebud':
    print('Access granted.')
if password != 'rosebud':
    print('Access denied.')
print('Done.')
```

عندما يتم تنفيذ السطر الذي يحوي على password = input() فإن البرنامج يتوقف وينتظر المستخدم ليقوم بعملية الإدخال، المستخدم يستطيع أن يدخل أي شيء يرديه والقيمة المدخلة سوف يتم حفظها في المتغير password

إذا كانت القيمة المدخلة هي 'rosebud' فإن نتيجة العبارة 'password == 'rosebud ستكون True (الشرط محقق) وعندها سيقوم البرنامج بتنفيذ الكتلة و طباعة العبارة 'Access granted.'

أما إذا كانت نتيجة العبارة 'password == 'rosebud هي False (الشرط غير محقق) فإن البرنامج لن ينفذ كتلة التعليمات التالية وينتقل إلى الشرط التالي 'password != 'rosebud والذي ستكون نتيجته True (محقق) وعندها سوف يقوم البرنامج بتنفيذ كتلة التعليمات التالية ويقوم بطباعة 'Access denied' أما إذا كانت نتيجة هذا الشرط هي False (غير محقق) فإن البرنامج سوف يتجاوز كتلة التعليمات.

التعليمة else:

عادةً نريد فحص حالة الشرط وتنفيذ كتلة من التعليمات إذا كان الشرط محقق وكتلة أخرى من التعليمات إذا كان الشرط غير محقق.

التعليمة else يمكن أن تستخدم بعد كتلة التعليمات الخاصة بتعليمه if وسوف يتم تنفيذ كتلة التعليمات الخاصة ب else في حال كان الشرط الخاص بتعليمه if غير محقق.

(إذا كان الشرط محقق نفذ هذه الكتلة أو إذا لم يكن محقق نفذ الكتلة التالية)

اكتب الكود التالي وقم حفظه باسم password2.py ولاحظ أنه يقوم بنفس العمل الذي يقوم به البرنامج الأول password.py ولكن في هذا البرنامج قمنا باستخدام if and else

Source code for password2.py

```
print('What is the password?')
password = input()
if password == 'rosebud':
    print('Access granted.')
else:
    print('Access denied.')
print('Done.')
```

التعليمة elif:

التعليمة elif هي اختصار ل else if

(إذا كان الشرط محقق نفذ كتلة التعليمات وإذا لم يكن محقق افحص حالة الشرط الثاني إذا كان محقق وإلا قم بتنفيذ أخر كتلة تعليمات)

اكتب البرنامج التالي في محرر النصوص وقم بحفظه باسم num.py

```
num = 12
if num < 10:
    print('الرقم مكون من خانة واحدة')
elif 9 < num < 100:
    print('الرقم مكون من خانتين')
elif 99 < num < 1000 :

    print('الرقم مكون من ثلاث خانات')
else:
    print('الرقم مكون من أكثر من ثلاث خانات')
```

ولأن الرقم هو 12 فسوف تكون نتيجة تنفيذ البرنامج هي:

```
>>>
الرقم مكون من خانتين
>>>
```

لاحظ أنه يتم تنفيذ كتلة واحدة فقط.

المعاملات in and not in:

العبارة التي تحوي على سلسلتين نصيتين مرتبطتين بالمعامل in سوف تعيد True إذا كانت السلسلة الأولى موجودة في السلسلة الثانية وإلا فسوف تعيد False.

انتبه أن المعاملات in , not in هي حساسة لحالة الأحرف.

جرب المثال التالي:

```
>>> 'hello' in 'hello world!'
True
>>> 'ello' in 'hello world!'
True
>>> 'HELLO' in 'hello world!'
False
>>> 'HELLO' in 'HELLO world!'
True
>>> '' in 'Hello'
True
>>> '' in ''
True
>>> 'D' in 'ABCDEF'
True
>>>
```

المعامل not in هو عكس المعامل in

جرب المثال التالي:

```
>>> 'hello' not in 'hello world!'
False
>>> 'ello' not in 'hello world!'
False
>>> 'HELLO' not in 'hello world!'
True
>>> 'Hello' not in 'HELLO world!'
True
>>> '' not in 'Hello'
False
>>> '' not in ''
False
>>> 'D' not in 'ABCDEF'
False
>>>
```

استخدام العبارات التي تحوي على المعاملات in , not in هو أمر مفيد جداً في الشرط الذي يلي تعليمة if

انتبه إلى أن كلمة in المستخدمة في تعليمة for (من أجل تحديد مجال الحلقة) ليست نفس المعامل in المستخدم هنا

الطريقة ()find:

()find هي طريقة خاصة بالسلاسل النصية string method

حيث يتم تمرير حرف لهذه الطريقة وتقوم هي بإعادة عدد صحيح هو دليل الفهرسة الذي يشير إلى مكان تواجد هذا الحرف داخل السلسلة النصية.

جرب المثال التالي:

```
>>> 'hello'.find('o')
4
>>> 'hello'.find('h')
0
>>> 'hello'.find('e')
1
>>> fizz = 'hello'
>>> fizz.find('h')
0
>>>
```

إذا تم تمرير حرف غير موجود في السلسلة النصية تقوم هذه الطريقة بإعادة القيمة 1-

الطريقة ()find هي حساسة لحالة الأحرف.

جرب المثال التالي:

```
>>> 'hello'.find('x')
-1
>>> 'hello'.find('H')
-1
>>>
```

القيمة التي يتم تمريرها للطريقة ()find يمكن أن تكون أكثر من حرف وسوف تعيد هذه الطريقة قيمة الفهرسة التي تدل على مكان أول حرف في السلسلة، جرب المثال التالي:

```
>>> 'hello'.find('x')
-1
>>> 'hello'.find('H')
-1
>>>
>>> 'hello'.find('ello')
1
>>> 'hello'.find('lo')
3
>>> 'hello hello'.find('e')
1
>>>
```

العودة إلى كود البرنامج:

الآن وبعد أن فهمت التعليمات if, elif, eles والمعاملات in, not in والطريقة find()

أصبح من السهل عليك فهم كيف يعمل برنامج شيفرة قيصر

```
caesarCipher.py
26.     if symbol in LETTERS:
27.         # get the encrypted (or decrypted) number for this symbol
28.         num = LETTERS.find(symbol) # get the number of the symbol
```

إذا كان الحرف الموجود في المتغير symbol (المتغير symbol الذي تستخدمه حلقة for) موجودة في السلسلة النصية المحفوظة في المتغير LETTERS (أي إذا كان هذا الحرف هو حرف كبير) فإن نتيجة العبارة symbol in LETTERS ستكون True (الشرط محقق)

تذكر أننا في السطر 22 قمنا بتحويل أحرف الرسالة إلى أحرف كبيرة باستخدام التعليمة

```
message = message.upper()
```

لذلك فإن القيمة symbol لا يمكن أن تكون حرف صغير

الحالة الوحيدة التي يكون فيها هذا الشرط غير محقق False هي عندما تكون قيمة symbol هي علامة ترقيم أو حرف مثل '4' or '?'

نريد أن نفحص قيمة symbol إذا كانت حرف كبير لأن برنامجنا سوف يقوم بتشفير أو فك تشفير الأحرف الكبيرة فقط، وأي أحرف أو رموز أخرى سوف تضاف إلى القيمة النصية المحفوظة في المتغير translator بدون تشفير أو فك تشفير.

كتلة تعليمات جديدة تبدأ بعد تعليمة if في السطر 26 وإذا نظرت إلى باقي البرنامج سوف تلاحظ أن كتلة التعليمة هذه تستمر إلى السطر 42 حيث توجد التعليمة else والتي هي جزء من التعليمة if الموجودة في السطر 26

caesarCipher.py

```
29.         if mode == 'encrypt':
30.             num = num + key
31.         elif mode == 'decrypt':
32.             num = num - key
```

الآن وبعد أن عرفنا رقم الحرف الموجود في المتغير num يمكننا تشفيره أو فك تشفيره من خلال إضافة أو طرح قيمة مفتاح التشفير من رقم هذا الحرف.

في شيفرة قيصر يتم إضافة قيمة مفتاح التشفير إلى رقم الحرف من أجل التشفير ويتم إنقاص قيمة مفتاح التشفير من رقم الحرف من أجل عملية فك التشفير.

المتغير mode يحوي على سلسلة نصية تخبر البرنامج أن يقوم بعملية التشفير أو فك التشفير إذا كانت قيمة هذا المتغير هي 'encrypt' فإن الشرط في السطر 29 سيكون محقق وسوف يقوم البرنامج بتنفيذ السطر 30 (وسوف يتجاوز كتلة التعليمات الخاصة بالتعليمة elif)

أما إذا كانت قيمة هذا المتغير هي أي قيمة أخرى غير 'encrypt' فسوف يكون الشرط في السطر 29 غير محقق False وعندها سوف يقوم البرنامج بتجاوز السطر 30 ويقوم بفحص حالة السطر الخاص بتعليمه elif في السطر 31

في هذه الطريقة يعرف البرنامج فيما إذا كان يجب عليه أن يقوم بعملية التشفير (إضافة قيمة مفتاح التشفير) أو أن يقوم بعملية فك التشفير (طرح قيمة مفتاح التشفير)

```
                                                              caesarCipher.py
34.         # handle the wrap-around if num is larger than the length of
35.         # LETTERS or less than 0
36.         if num >= len(LETTERS):
37.             num = num - len(LETTERS)
38.         elif num < 0:
39.             num = num + len(LETTERS)
```

تذكر عندما قمنا بتنفيذ تشفير قيصر باستخدام الورقة والقلم في الحالات التي كانت نتيجة جمع قيمة المفتاح مع العدد الخاص بالحرف المراد تشفيره وكانت النتيجة أكبر من 26 أو في حالة فك التشفير عند القيام بعملية الطرح وكانت النتيجة أصغر من 0 ، في هذه الحالات كنا نقوم بجمع أو طرح القيمة 26 إلى أو من النتيجة من أجل الحصول على القيمة الصحيحة وهذه العملية تتم في الأسطر من 36 to 39 من البرنامج.

إذا كانت قيمة num أكبر أو تساوي 26 فإن الشرط في السطر 36 سيكون محقق True وسوف يتم تنفيذ السطر 37 (ويتم تجاوز التعليمة elif في السطر 38)

ومن ناحية أخرى فإن بايثون يقوم بفحص إذا كانت قيمة num أصغر من 0 فإذا كان هذا الشرط محقق فسوف يتم تنفيذ السطر 39

في شيفرة قيصر يتم إضافة أو طرح القيمة 26 لأن عدد الأحرف في اللغة الإنكليزية هو 25 حرف.

لاحظ أنه بدلاً من استخدام القيمة 26 بشكل مباشر قمنا باستدعاء التابع len(LETTERS) والذي سوف يقوم بإعادة القيمة 26

يمكننا تعديل القيمة المحفوظة في المتغير LETTERS من أجل تشفير أو فك تشفير أحرف غير كبيرة وهذا ما سيتم شرحه في نهاية هذا الفصل

```
                                                              caesarCipher.py
        # add encrypted/decrypted number's symbol at the end of translated
        translated = translated + LETTERS[num]
```

الآن وبعد أن تم تعديل قيمة العدد الموجود في المتغير num وأصبح قيمة الفهرسة أو دليل الحرف المراد تشفيره أو فك تشفيره، نريد إضافة الأحرف المشفرة أو غير مشفرة إلى نهاية السلسلة النصية المحفوظة في المتغير translated

```
caesarCipher.py
44.     else:
45.         # just add the symbol without encrypting/decrypting
46.         translated = translated + symbol
```

في السطر 44 التعليمة else تبعد مسافة بدائية مقدارها أربعة فراغات وهذا يدل على أنها استمرار للتعليمة if الموجودة في السطر 26

إذا كانت الشرط الخاص بتعليمه if في السطر 26 غير محقق فإن كتلة التعليمات السابقة سوف يتم تجاوزها والانتقال إلى تنفيذ التعليمة الخاصة ب else في السطر 46

(السطر 45 يتم تجاوزه لأنه عبارة عن تعليق)

كتلة التعليمات الخاصة ب else هي عبارة عن سطر واحد حيث يتم إضافة القيمة المحفوظة في المتغير symbol إلى القيمة الموجودة في المتغير translated

الرموز أو القيم الغير حرفية مثل الفراغ ' ' أو النقطة ' . ' يتم إضافتها إلى translated بدون أي تشفير أو فك تشفير.

عرض ونسخ النص المشفر أو الغير مشفر:

```
caesarCipher.py
48. # print the encrypted/decrypted string to the screen
49. print(translated)
50.
51. # copy the encrypted/decrypted string to the clipboard
52. pyperclip.copy(translated)
```

السطر 49 لا يحوي على مسافة بدائية وهذا يعني أن كتلة التعليمات السابقة قد انتهت

في السطر 49 تم استدعاء التابع ()print من أجل أن يقوم بطباعة قيمة المتغير translated

في السطر 52 تم استدعاء التابع الموجود في الوحدة pyperclip module والمسمى copy() وحيث يتم تمرير قيمه له ليتم نسخها إلى الذاكرة لنتمكن من لصقها في مستند نصي أو في أي مكان أخر.

إذا قمت باستدعاء التابع بالشكل التالي copy(translated) بدلاُ من الشكل pyperclip.copy(translated) فسوف تظهر رسالة خطأ

```
>>> copy('Hello')
Traceback (most recent call last):
  File "<pyshell#92>", line 1, in <module>
    copy('Hello')
NameError: name 'copy' is not defined
>>>
```

وفي حال نسيت استدعاء الوحدة import pyperclip فلن تتمكن من استدعاء التابع pyperclip.copy() وسوف تحصل على رسالة خطأ.

هذا هو برنامج شيفرة قيصر، عندما تريد تشغيله لاحظ أن جهاز الكمبيوتر يمكنه تنفيذ كامل البرنامج وتشفير السلسلة النصية بأجزاء من الثانية، حتى ولو كتبت رسالة طويلة جداً داخل المتغير message فإن جهاز الكمبيوتر قادر على تشفيرها أو فك تشفيرها بأقل من ثانيتين

الآن أصبح بإمكانك تشفير أو فك تشفير أي رسالة بشيفرة قيصر ونسخها ولصقها في أي مكان أو أي ملف أو ارسالها عبر الايميل.

تشفير القيم الغير حرفية:

برنامج شيفرة قيصر الذي قمنا بكتابته لا يستطيع تشفير القيم الغير حرفية.

إذا كنت تريد تشفير السلسلة النصية التالية 'The password is 31337' باستخدام مفتاح التشفير 20 فسوف تتم عملية التشفير وسوف نحصل على السلسلة المشفرة التالية

'Dro zkccgybn sc 31337' هذه الرسالة المشفرة لا تحافظ على سرية الرسالة ويمكننا تعديل البرنامج السابق من أجل تشفير القيم الغير حرفية.

إذا قمت بتغير السلسلة النصية المحفوظة في المتغير LETTERS لتحوي على قيم غير الأحرف الكبيرة وذلك لأن السطر 26 يحوي على الشرط symbol in LETTERS والذي يجب أن يكون محقق.

بعد تعديل الكود سوف تعرف السبب من استخدام len(LETTERS) بدلاً من كتابة 26 بشكل مباشر.

تعديل الكود يتم من خلال تعديل السطر 16 ليصبح يحوي على رموز وأرقام وجعل السطر 22 عبارة عن تعليق عن طريق إضافة الرمز # في بداية هذا السطر.

يمكنك الحصول على الكود الجديد للبرنامج من http://invpy.com/caeserCipher2.py

لقد قمت أنا بالتعديل على البرنامج لتتمكن من ادخال النص وقيمة المفتاح ونوع العملية اثناء تنفيذ البرنامج وذلك من خلال تعديل الأسطر التالية لتصبح كالتالي:

```
7.  message = input('input text : '  , )

10. key = input('input key 0 - 26 : ' ,)

30. num = num + int(key)

32. num = num - int(key)
```

ليصبح الكود البرمجي بشكل كامل بعد التعديل كالتالي:

```
# شيفرة قيصر

import pyperclip
message = input('input text: ' ,)
key = input('input key 0 - 26 :' ,)
mode = input('chose encrypt or decrypt: ' ,)
LETTERS = '!"#$%&\'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQ
translated = ''
for symbol in message:
    if symbol in LETTERS:
        num = LETTERS.find(symbol)
        if mode == 'encrypt':
            num = num + int(key)
        elif mode == 'decrypt':
            num = num - int(key)
        if num >= len(LETTERS):
            num = num - len(LETTERS)
        elif num < 0:
            num = num + len(LETTERS)
        translated = translated + LETTERS[num]
    else:
        translated = translated + symbol
print(translated)
pyperclip.copy(translated)
```

هذا الكود الأخير وبعد التعديل غير موجود في موقع الكتاب باللغة الأجنبية وسوف أقوم بإرفاقه مع النسخة العربية

الآن وبعد التعديل يتم تنفيذ البرنامج بالشكل التالي من أجل القيام بعملية التشفير

```
>>>
input text: syria
input key 0 - 26 :2
chose encrypt or decrypt: encrypt
u{tkc
```

من أجل القيام بعملية فك التشفير

```
input text: u{tkc
input key 0 - 26 :2
chose encrypt or decrypt: decrypt
syria
>>> .
```

عند التنفيذ يطلب البرنامج من المستخدم إدخال النص وبعد إدخال النص والضغط على انتر يطلب البرنامج من المستخدم إدخال مفتاح تشفير له قيمة من 26 – 2 وبعد إدخال قيمة مفتاح التشفير والضغط على انتر يطلب البرنامج تحديد نوع العملية (تشفير أو فك تشفير) وبهذا الشكل يمكن استخدام البرنامج للقيام بعمليات تشفير وفك تشفير لنصوص مختلفة وباختيار قيم مختلفة لمفتاح التشفير بشكل سهل وبدون التعديل على الكود البرمجي في كل مرة.

الخلاصة:

الآن لقد تعلمت مبادئ البرمجة الأساسية وأصبحت قادراً على كتابة برنامج يقوم بالتشفير وفك التشفير والأمر الأهم أنك أصبحت قادراً على فهم كيف يعمل كود البرنامج.

الوحدة module هي برنامج بلغة البايثون يحوي على توابع مفيدة يمكنك استخدامها.

من أجل استخدام هذه التوابع يجب عليك أولاً استيراد الوحدة باستخدام التعليمة import.

استدعاء التوابع الموجودة داخل الوحدة يتم من خلال كتابة اسم الوحدة مسبوقاً بنقطة قبل اسم التابع كما في المثال التالي: module.function()

الطريقة method هي عبارة عن تابع يلحق بقيمة معينة من نوع معين من البيانات.

الطريقتين upper() and lower() هما طريقتين نصيتين string methods ويقومان بإعادة أحرف كبيرة أو أحرف صغيرة للسلسلة النصية التي تقوم باستدعائهما.

الطريقة find() هي أيضاً طريقة نصية string method والتي تقوم بإعادة قيمة عدد صحيح يدل على عنوان الفهرسة لمكان وجود الحرف أو القيمة الممررة إليها في السلسلة النصية.

حلقة for هي حلقة تقوم بتكرار الأحرف الموجود داخل السلسلة وتقوم بتنفيذ التعليمات داخل الكتلة التالية من أجل كل دورة.

التعليمات if, elif, eles تقوم بتنفيذ كتلة من التعليمات بالاعتماد على حالة الشرط إذا كان محقق أو غير محقق.

المعاملان in , not in يستخدمان من أجل معرفة وجود أو عدم وجود حرف أو سلسلة نصية داخل سلسلة نصية أخرى ويعيدان إحدى القيمتين True or False.

في الفصل القادمة سوف نستخدم هذه المهارات من أجل كتابة برنامج يقوم بفك شيفرة قيصر وقراءة النص الصريح لرسائل مشفرة بشيفرة قيصر.



فك شيفرة قيصر باستخدام تقنية القوة الغاشمة

محتوى هذا الفصل:

- قول شانون المأثور وقاعدة كيرشوف
- تقنية القوة الغاشمة brute-force
- التابع ()range
- تنسيق السلاسل النصية

فك الشيفرات Hacking Ciphers:

يمكن فك شيفرة قيصر باستخدام تقنية تحليل التشفير: القوة الغاشمة brute-force

لأن البرنامج الذي سنقوم بكتابته قادر على كسر تشفير قيصر فلا تستخدم هذه الشيفرة في تشفير معلوماتك السرية.

قاعدة كيرشوف (نسبة لعالم التشفير اجوست كيرشوف) والتي تقول: " نظام التشفير يجب أن يبقى سرياً حتى ولو عرف كل شخص هذا النظام، باستثناء المفتاح الذي يكون معرف من قبل الناس

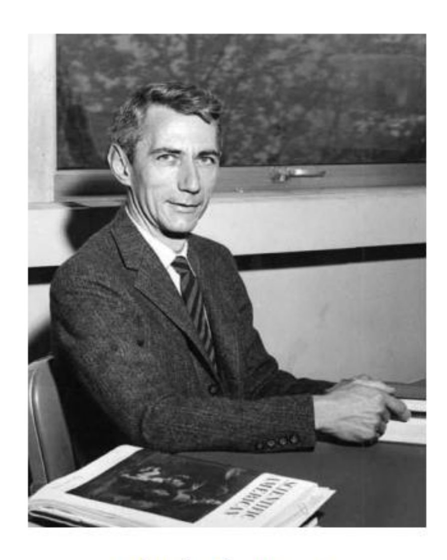


Auguste Kerckhoffs January 19, 1835 - August 9, 1903

"A cryptosystem should be secure even if everything about the system, except the key, is public knowledge."

قول مأثور لعالم الرياضيات كلويد شانون:

" العدو يعرف النظام"



Claude Shannon
April 30, 1916 - February 24, 2001

"The enemy knows the system."

هجوم القوة الغاشمة:

لا يوجد أحد يستطيع منع محلل الشيفرات من تخمين قيمة مفتاح التشفير، وفك تشفير النص المشفر باستخدام هذا المفتاح وإذا لم يكن هذا المفتاح هو المفتاح الصحيح سوف يحاول استخدام قيمة أخرى ويستمر باستخدام قيم أخرى حتى الوصول الي القيمة الصحيحة.

التقنية التي يتم من خلالها استخدام كل قيم الممكنة لمفتاح التشفير تسمى هجوم القوة الغاشمة brute-force attack وهذا الهجوم ليس هجوم معقد وهو غير فعال في كل الشيفرات ولكنه فعال ضد شيفرة قيصر.

النص البرمجي لبرنامج فك شيفرة قيصر:

افتح نافذة محرر النصوص من خلال File > New File وقم بكتابة الكود التالي ثم قم بحفظه باسم caesarHacker.py واضغط F5 من أجل تنفيذ هذا البرنامج

يجب أن تقوم بتحميل الوحدة (الموديول) pyperclip.py module ووضعها في نفس المجلد الذي يحوي على الملف caesarHacker.py

يمكنك الحصول على هذه الوحدة من http://invpy.com/pyperclip.py

Source code for caesarHacker.py

```python
# Caesar Cipher Hacker
# http://inventwithpython.com/hacking (BSD Licensed)

message = 'GUVF VF ZL FRPERG ZRFFNTR.'
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'

# loop through every possible key
for key in range(len(LETTERS)):

    # It is important to set translated to the blank string so that the
    # previous iteration's value for translated is cleared.
    translated = ''

    # The rest of the program is the same as the original Caesar program:

    # run the encryption/decryption code on each symbol in the message
    for symbol in message:
        if symbol in LETTERS:
            num = LETTERS.find(symbol) # get the number of the symbol
            num = num - key

            # handle the wrap-around if num is 26 or larger or less than 0
            if num < 0:
                num = num + len(LETTERS)

            # add number's symbol at the end of translated
            translated = translated + LETTERS[num]

        else:
            # just add the symbol without encrypting/decrypting
            translated = translated + symbol

    # display the current key being tested, along with its decryption
    print('Key #%s: %s' % (key, translated))
```

وعند تشغيل هذا البرنامج سوف نحصل على النتيجة التالية:

```
Key #0: GUVF VF ZL FRPERG ZRFFNTR.
Key #1: FTUE UE YK EQODQF YQEEMSQ.
Key #2: ESTD TD XJ DPNCPE XPDDLRP.
Key #3: DRSC SC WI COMBOD WOCCKQO.
Key #4: CQRB RB VH BNLANC VNBBJPN.
Key #5: BPQA QA UG AMKZMB UMAAIOM.
Key #6: AOPZ PZ TF ZLJYLA TLZZHNL.
Key #7: ZNOY OY SE YKIXKZ SKYYGMK.
Key #8: YMNX NX RD XJHWJY RJXXFLJ.
Key #9: XLMW MW QC WIGVIX QIWWEKI.
Key #10: WKLV LV PB VHFUHW PHVVDJH.
Key #11: VJKU KU OA UGETGV OGUUCIG.
Key #12: UIJT JT NZ TFDSFU NFTTBHF.
Key #13: THIS IS MY SECRET MESSAGE.
Key #14: SGHR HR LX RDBQDS LDRRZFD.
Key #15: RFGQ GQ KW QCAPCR KCQQYEC.
Key #16: QEFP FP JV PBZOBQ JBPPXDB.
Key #17: PDEO EO IU OAYNAP IAOOWCA.
Key #18: OCDN DN HT NZXMZO HZNNVBZ.
Key #19: NBCM CM GS MYWLYN GYMMUAY.
Key #20: MABL BL FR LXVKXM FXLLTZX.
Key #21: LZAK AK EQ KWUJWL EWKKSYW.
Key #22: KYZJ ZJ DP JVTIVK DVJJRXV.
Key #23: JXYI YI CO IUSHUJ CUIIQWU.
Key #24: IWXH XH BN HTRGTI BTHHPVT.
Key #25: HVWG WG AM GSQFSH ASGGOUS.
>>>
```

لاحظ أن نتيجة فك التشفير باستخدام المفتاح 13 هي العبارة المفهومة وبالتالي فإن مفتاح التشفير المستخدم في عملية التشفير يجب أن يكون 13

كيف يعمل هذا البرنامج:

caesarHacker.py

```
# Caesar Cipher Hacker
# http://inventwithpython.com/hacking (BSD Licensed)

message = 'GUVF VF ZL FRPERG ZRFFNTR.'
LETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
```

هذا البرنامج يبدأ بخلق متغير باسم message والذي يحوي على النص المشفر المراد فك تشفيره.

المتغير LETTERS يحوي على كل الأحرف التي يمكن أن يتم تشفيرها باستخدام الشيفرة وهذا المتغير يجب أن يحوي على نفس القيم التي يحويها المتغير LETTERS الموجود في برنامج شيفرة قيصر الذي استخدم من أجل القيام بعملية التشفير وإلا فإن برنامج فك الشيفرة لن يعمل بشكل صحيح.

التابع ()range:

caesarHacker.py

```
7. # loop through every possible key
8. for key in range(len(LETTERS)):
```

السطر 8 يحوي على حلقة for والتي تقوم بالدوران عبر القيم التي يعيدها التابع ()range

التابع ()range يأخذ قيمة عدد صحيح ويعيد قيم لمجال هذه القيمة وقيم هذا المجال يمكن أن تستخدم في خلقة for من أجل تكرار الحلقة عدد معين من المرات

جرب المثال التالي:

```
>>> for i in range(4):
	print(i)


0
1
2
3
>>>
```

```
>>> for i in range(4):
	print('Hello')


Hello
Hello
Hello
Hello
>>>
```

في السطر 8 حلقة for تقوم بضبط قيمة المفتاح ضمن المجال من 26 – 0 (ولكن لا تتضمن 26) بدل من أن نقوم بكتابة القيمة 26 بشكل مباشر في كود البرنامج، قمنا باستخدام القيمة المعادة من التابع len(LETTERS) والسبب في ذلك هو في حال إضافة قيم جديد أو تعديل القيم الموجودة في المتغير LETTERS سيبقى البرنامج يعمل بشكل صحيح (عد إلى تشفير القيم الغير حرفية في الفصل السابق لمعرفة السبب)

عندما يتم تنفيذ البرنامج أول مرة فهو يذهب عبر الحلقة ويتم ضبط قيمة المفتاح key = 0 ويتم فك تشفير الرسالة باستخدام المفتاح 0 (الكود الموجود ضمن حلقة for هو الذي يقوم بعملية فك التشفير) وعند تنفيذ الدورة الثانية للحلقة for في السطر الثامن فسوف يتم ضبط قيمة المفتاح key = 1 ويتم فك تشفير الرسالة بالمفتاح 1 وتستمر هذه العملية إلى أن نصل إلى قيمة المفتاح 26

يمكنك تمرير عددين صحيحين للتابع range() بدل قيمة واحدة، حيث أن القيمة الأولى هي بداية المجال والقيمة الثانية هي لنهاية المجال ويتم الفصل بين القيمتين بفاصلة ' , '

جرب المثال التالي:

```
>>> for i in range(2,6):
	print(i)


2
3
4
5
>>>
```

العودة إلى كود البرنامج:

```
caesarHacker.py
7. # loop through every possible key
8. for key in range(len(LETTERS)):
9.
10.     # It is important to set translated to the blank string so that the
11.     # previous iteration's value for translated is cleared.
12.     translated = ''
```

في السطر 12 تم إسناد قيمة سلسلة نصية فارغة للمتغير translated وفي الأسطر القادمة سوف يتم إضافة النص بعد عملية فك التشفير إلى المتغير translated

```
caesarHacker.py
14.     # The rest of the program is the same as the original Caesar program:
15.
16.     # run the encryption/decryption code on each symbol in the message
17.     for symbol in message:
18.         if symbol in LETTERS:
19.             num = LETTERS.find(symbol) # get the number of the symbol
```

من السطر 17 – 31 استخدمنا نفس الكود الذي استخدمناه في برنامج شيفرة قيصر في الفصل السابق ولكنه مختلف قليلاً لأن هذا الكود يقوم بعملية فك التشفير فقط بدلاً من التشفير وفك التشفير

في السطر 17 قمنا بتعريف حلقة for تقوم بالمرور أو الدوران على كل رمز من السلسلة النصية المشفرة الموجودة في المتغير message

في السطر 18 يتم فحص كل رمز فيما إذا كان حرف كبير (إذا كان موجود ضمن المتغير LETTERS والذي يحوي على أحرف كبيرة فقط) إذا كان هذا محقق فيتم فك تشفير هذا الرمز.

في السطر 19 يتم تحديد مكان توضع الرمز في المتغير LETTERS باستخدام الطريقة find() ويتم تخزين القيمة التي تدل على مكان هذا الرمز في المتغير num

```
caesarHacker.py
20.                 num = num - key
21.
22.                 # handle the wrap-around if num is 26 or larger or less than 0
23.                 if num < 0:
24.                     num = num + len(LETTERS)
```

في السطر 20 يتم طرح قيمة المفتاح من قيمة num (في شيفرة قيصر يتم الطرح في حالة فك التشفير) وفي حالة كانت النتيجة أصغر من الصفر يتم إضافة القيمة 26 (القيمة المعادة من التابع (len(LETTERS)

```
caesarHacker.py
26.                 # add number's symbol at the end of translated
27.                 translated = translated + LETTERS[num]
```

الآن وبعد أن تم تعديل قيمة num يتم حساب قيمة LETTERS[num] والتي هي الحرف أو الرمز بعد عملية فك التشفير

في السطر 27 يتم إضافة هذا الرمز إلى السلسلة النصية الموجودة في المتغير translated

```
caesarHacker.py
29.             else:
30.                 # just add the symbol without encrypting/decrypting
31.                 translated = translated + symbol
```

إذا كان الشرط في السطر 18 غير محقق أي أن الرمز غير موجود في LETTERS (ليس حرف كبير) فلن نطبق عليه عملية فك التشفير ويتم إضافته كما هو إلى القيمة النصية الموجودة في المتغير translated

التعليمة else هي خاصة بالتعليمة if في السطر 18

تنسيق السلسلة النصية:

```
caesarHacker.py
33.     # display the current key being tested, along with its decryption
34.     print('Key #%s: %s' % (key, translated))
```

في السطر 34 قمنا باستدعاء التابع print() ليقوم بطباعة عدة أسطر بحسب عدد مرات تكرار الحلقة for في السطر 8

القيم التي تم تمريرها إلى التابع print() لم نستخدمها من قبل، هذه القيم تقوم بتنسيق السلسلة النصية.

القيمة s% تستخدم من أجل وضع سلسلة نصية بجانب سلسلة نصية أخرى، حيث يتم استبدال s% الأولى بقيمة المتغير الأول واستبدال s% الثانية بقيمة المتغير الثاني

جرب المثال التالي:

```
>>> 'Hello %s!' % ('world')
'Hello world!'
>>> 'Hello' + ' world' + '!'
'Hello world!'
>>> 'The %s ate the %s that ate %s.' % ('dog', 'cat', 'rat')
'The dog ate the cat that ate rat.'
>>>
```

تنسيق النصوص بهذه الطريقة هو أسهل من عملية جمع النصوص باستخدام المعامل '+'

إحدى ميزات تنسيق النص هذه هي إمكانية إضافة قيم غير نصية (أرقام مثلاً) إلى السلسلة النصية وهذا الأمر غير ممكن باستخدام المعامل '+' حيث لا يمكن جمع قيمة نصية مع قيمة رقمية وسوف تحصل على رسالة خطأ، جرب المثال التالي:

```
>>> '%s had %s pies.' % ('Alice', 42)
'Alice had 42 pies.'
>>> 'Alice' + ' had ' + 42 + ' pies.'
Traceback (most recent call last):
  File "<pyshell#8>", line 1, in <module>
    'Alice' + ' had ' + 42 + ' pies.'
TypeError: Can't convert 'int' object to str implicitly
>>>
```

في السطر 34 من كود البرنامج استخدمنا خاصية تنسق النص من أجل خلق سلسلة نصية تحوي على قيمتي المتغيرين key, translated لأن المتغير key يحوي على قيمة عددية (عدد صحيح) أما المتغير translated يحوي على سلسلة نصية لذلك استخدمنا خاصية التنسيق السابقة.

الخلاصة:

الضعف في شيفرة قيصر هو وجود عدد محدود من المفاتيح التي يمكن أن تستخدم في عملية التشفير وفك التشفير

أي جهاز كمبيوتر يمكنه بسهولة القيام بعملية فك التشفير باستخدام ال 26 مفتاح المحتملين ثم يقوم محلل الشيفرة بالنظر إلى الرسائل الناتجة من عملية فك التشفير ومعرفة الرسالة الحقيقية.

من أجل جعل رسائل أكثر سرية يجب أن تستخدم عدد أكبر من مفاتيح التشفير.

شيفرة التحويل في الفصل القادم يمكن أن تؤمن هذا لك.



التشفير باستخدام شيفرة التحويل

محتوى هذا الفصل:

- خلق تابع باستخدام التعليمة def
- التابع ()main
- البارامترات
- المجالات العامة والمجالات المحلية والمتغيرات العامة والمتغيرات المحلية
- التابع ()list
- المعاملات (=/ ,=* ,=- ,=+)
- الطريقة النصية ()join
- إعادة القيم والتعليمة return
- المتغير الخاص _name_

شيفرة قيصر ليست شيفرة آمنة لأنه يمكن فكها باستخدام تقنية القوة الغاشمة من خلال المرور على كل القيم الممكنة ال 26 لمفتاح التشفير.

شيفرة التحويل تملك عدد أكبر من مفاتيح التشفير الممكنة وهذا يجعل هجوم القوة الغاشمة أكثر صعوبة.

التشفير باستخدام شيفرة التحويل:

Encryption with the Transposition Cipher

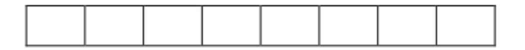
بدلاً من استبدال الحروف بحروف أخرى فإن شيفرة التحويل تقوم بدمج رموز في نص الرسالة من أجل جعل الرسالة الأصلية غير قابلة للقراءة.

قبل البدء بكتابة الكود، لنقم بتشفير الرسالة 'Common sense is not so common.'

باستخدام الورقة والقلم وبشكل يتضمن الفراغات وعلامات الترقيم، هذا يعني أن الرسالة مكونة من 30 حرف.

سوف نستخدم الرقم 8 كمفتاح.

أول خطوة هي رسم عدد من الصناديق مساوي لقيمة المفتاح (ثمانية صناديق في هذا المثال)



ثاني خطوة هي كتابة الرسالة المراد تشفيرها في داخل الصناديق، كل حرف في صندوق

تذكر أن المسافة الفارغة تعتبر حرف (في هذا الكتاب قمنا بكتابة (S) للإشارة إلى المسافة الفارغة لكي لا تبقى صناديق فارغة

C	o	m	m	o	n	(s)	s

نحن نملك فقط 8 صناديق ولكن الرسالة المراد تشفيرها مكونة من 30 حرف لذلك سوف نقوم برسم ثمانية صناديق أخرى تحت الصناديق الأولى ونقوم بكتابة بقية الأحرف في داخلها

1st	2nd	3rd	4th	5th	6th	7th	8th
C	o	m	m	o	n	(s)	s
e	n	s	e	(s)	i	s	(s)
n	o	t	(s)	s	o	(s)	c
o	m	m	o	n	.		

قمنا بتظليل آخر صندوقين من أجل تجاهلهم.

النص المشفر هو الأحرف التي تقرأ من الأعلى إلى الأسفل لكل عمود

'C' , 'e' , 'n' , 'o' هذه الأحرف من العمود الأول بعدها ننتقل لأحرف العمود الثاني ثم الثالث وهكذا حتى نهاية العواميد

النص المشفر هو قراءة الأحرف من الأعلى إلى الأسفل ثم الانتقال إلى العمود الثاني

'o' , 'n' , 'o' , 'm'

ويتم تجاهل الصناديق المظللة.

النص المشفر هو 'Cenoonommstmme oo snnio s s c'

وهو ممزوج بشكل كاف لمنع أي شخص من معرفة نص الرسالة الأصلية من خلال النظر إلى الرسالة المشفرة.

خطوات عملية التشفير:

١. تحديد عدد أحرف الرسالة واختيار قيمة مفتاح التشفير
٢. رسم عدد من الصناديق مساوي لقيمة مفتاح التشفير
٣. ملئ الصناديق بأحرف الرسالة من اليسار إلى اليمين
٤. عند امتلأ الصناديق وعدم انتهاء أحرف الرسالة يتم إضافة سطر جديد من الصناديق
٥. تظليل الصناديق الفارغة في نهاية الصناديق
٦. كتابة الرسالة المشفرة من خلال قراءة الأحرف من الأعلى إلى الأسفل وعند الانتهاء من أول عمود ننتقل إلى العمود المجاور ويتم تجاهل الصناديق المظللة

برنامج التشفير باستخدام شيفرة التحويل:

التشفير باستخدام الورق والقلم يتطلب الكثير من الجهد ويمكن أن نقع بأخطاء كتابية أثناء عملية التشفير.

لنبدأ بكتابة برنامج يقوم بعملية التشفير باستخدام شيفرة التحويل (برنامج فك التشفير سوف يتم شرحه في الفصل القادم)

عند استخدام هذا البرنامج سوف تظهر مشكلة بسيطة، إذا كانت المسافات الفارغة في نهاية النص المشفر فلن نتمكن من رؤية هذه المسافات أو لن نتمكن من معرفة عدد هذه المسافات ومن أجل حل هذه المشكلة فإن البرنامج سوف يضيف الرمز | في نهاية النص المشفر (الرمز | موجود في لوحة المفاتيح فوق زر انتر)

مثلاً:

```
Hello|   # There are no spaces at the end of the message.
Hello |  # There is one space at the end of the message.
Hello  | # There are two spaces at the end of the message.
```

الكود البرمجي لبرنامج الشفير باستخدام شيفر التحويل:

قم بفتح ملف جديد من خلال File > New File ثم قم بكتابة الكود التالي في داخله ثم قم بحفظه باسم transpositionEncrypt.py واضغط F5 من أجل تنفيذ هذا البرنامج

يجب أن تقوم أولاُ بوضع الوحدة pyperclip.py في نفس المجلد الذي سوف تقوم بحفظ البرنامج في داخله، يمكنك الحصول على pyperclip.py module من

http://invpy.com/pyperclip.py

Source code for transpositionEncrypt.py

```
# Transposition Cipher Encryption
# http://inventwithpython.com/hacking (BSD Licensed)

import pyperclip

def main():
    myMessage = 'Common sense is not so common.'
    myKey = 8

    ciphertext = encryptMessage(myKey, myMessage)

    # Print the encrypted string in ciphertext to the screen, with
    # a | (called "pipe" character) after it in case there are spaces at
    # the end of the encrypted message.
    print(ciphertext + '|')

    # Copy the encrypted string in ciphertext to the clipboard.
    pyperclip.copy(ciphertext)


def encryptMessage(key, message):
    # Each string in ciphertext represents a column in the grid.
    ciphertext = [''] * key

    # Loop through each column in ciphertext.
    for col in range(key):
        pointer = col

        # Keep looping until pointer goes past the length of the message.
        while pointer < len(message):
            # Place the character at pointer in message at the end of the
            # current column in the ciphertext list.
            ciphertext[col] += message[pointer]

            # move pointer over
            pointer += key

    # Convert the ciphertext list into a single string value and return it.
    return ''.join(ciphertext)


# If transpositionEncrypt.py is run (instead of imported as a module) call
# the main() function.
if __name__ == '__main__':
    main()
```

نتيجة تنفيذ هذا البرنامج هي:

```
Cenoonommstmme oo snnio. s s c|
>>>
```

النص المشفر (بدون الرمز |) تم نسخه إلى الذاكرة ويمكنه لصقه في أي ملف نصي أو في رسالة إيميل لتقوم بإرساله إلى أي شخص تريد.

إذا كنت تريد تشفير رسالة مختلفة أو استخدام مفتاح تشفير مختلف قم بتغير القيم المسندة للمتغيرات myMessage and myKey في السطرين 7 and 8

كيف يعمل هذا البرنامج:

```
transpositionEncrypt.py
1. # Transposition Cipher Encryption
2. # http://inventwithpython.com/hacking (BSD Licensed)
3.
4. import pyperclip
```

في برنامج التشفير باستخدام شيفرة التحويل وبشكل مماثل لبرنامج شيفرة قيصر سيتم نسخ النص المشفر إلى الذاكرة لذلك قمنا باستيراد الوحدة pyperclip لنتمكن من استدعاء التابع pyperclip.copy()

خلق تابع باستخدام التعليمة def:

```
transpositionEncrypt.py
6. def main():
7.     myMessage = 'Common sense is not so common.'
8.     myKey = 8
```

التابع (مثل التابع ()print) هو برنامج صغير داخل البرنامج الأصلي

عندما يتم استدعاء التابع يقوم البرنامج بتنفيذ التعليمات الموجودة داخل التابع ثم يعود لتنفيذ الكود في السطر الذي يلي استدعاء التابع.

يمكنك خلق تابع باستخدام التعليمة def كما في السطر 6

التعليمة def في السطر 6 لا تقوم باستدعاء التابع المسمى ()main ولكن تقوم بخلق تابع جديد اسمه ()main وهذه التعليمة هي اختصار للكلمة defining (تعريف).

بعد خلق التابع يمكننا استدعائه لاحقاً في برنامجك وعند الاستدعاء فإن البرنامج سوف ينتقل لتنفيذ الكود الموجود داخل هذا التابع (كتلة التعليمات التي تلي التعليمة def)

افتح نافذة جديدة لمحرر النصوص واكتب الكود التالي:

Source code for helloFunction.py

```
def hello():
    print('Hello!')
    total = 42 + 1
    print('42 plus 1 is %s' % (total))
print('Start!')
hello()
print('Call it again.')
hello()
```

قم بحفظه باسم helloFunction.py ثم اضغط F5 من أجل التنفيذ وستكون النتيجة كالتالي:

```
>>>
Start!
Hello
42 plus 1 is 43
Call it again.
Hello
42 plus 1 is 43
>>>
```

عندما يعمل هذا البرنامج فهو يبدأ التنفيذ من الأعلى، في أول سطر التعليمة def التي تقوم بتعريف تابع باسم ()hello ثم يقوم البرنامج بتجاوز تنفيذ كتلة التعليمات التي تلي التعليمة def

ويقوم بتنفيذ التعليمة print('Start!') وهذا سبب ظهور الكلمة 'Start!' في أول سطر من نتيجة تنفيذ البرنامج.

في السطر الذي يلي التعليمة print('Start!') يتم استدعاء التابع hello() وعندها فإن البرنامج سوف ينتقل لتنفيذ الكود الموجود في كتلة التعليمات الخاصة بهذا التابع.

هذا التابع يقوم بطباعة الكلمة 'Hello' ويقوم بجمع 1 + 42 ويطبع النتيجة على الشاشة.

عندما ينتهي البرنامج من تنفيذ التعليمات الخاصة بالتابع فسوف يعود لتنفيذ ما تبقى من الكود الموجود في السطر الذي يلي تعليمة استدعاء التابع وهو print('Call it again.') وبعد تنفيذ هذه التعليمة يتم استدعاء التابع hello() مرة ثانية وعندها سوف ينتقل البرنامج لتنفيذ الكود الخاص بالتابع مرة ثانية.

التابع main() في البرنامج:

transpositionEncrypt.py

```
6. def main():
7.     myMessage = 'Common sense is not so common.'
8.     myKey = 8
```

في برامج هذا الكتاب يوجد تابع باسم main() والذي عند استدعاءه يقوم بتشغيل البرنامج.

السبب سوف يتم شرحه في نهاية هذا الفصل ولكن الآن يجب أن تعرف أن التابع main() في البرامج في هذا الكتاب دائماً يتم استدعاءه بعد أن تعمل البرامج.

في السطرين 7 and 8 تم تعريف المتغيرين myMessage and myKey حيث سيتم تخزين النص الصريح (النص الغير مشفر) ومفتاح التشفير في هذين المتغيرين

transpositionEncrypt.py

```
    ciphertext = encryptMessage(myKey, myMessage)
```

الكود الذي سوف يقوم بعملية التشفير قمنا بوضعه داخل تابع يتم تعريفه في السطر 21 باسم encryptMessage()

هذا التابع سوف يمرر له قيمتين:

- قيمة عددية (عدد صحيح) وهي قيمة المفتاح
- وقيمة نصية وهي الرسالة المراد تشفيرها

عند استدعاء هذا التابع يتم الفصل بين القيم الممررة إلية باستخدام فاصلة ' , '

القيمة التي يعيدها التابع encryptMessage() هي قيمة نصية وهي النص المشفر.

كود هذا التابع سوف يتم شرحه لاحقاً في هذا الفصل.

القيم التي يعيدها هذا التابع سوف يتم تخزينها في المتغير ciphertext

transpositionEncrypt.py

```
12.     # Print the encrypted string in ciphertext to the screen, with
13.     # a | (called "pipe" character) after it in case there are spaces at
14.     # the end of the encrypted message.
15.     print(ciphertext + '|')
16.
17.     # Copy the encrypted string in ciphertext to the clipboard.
18.     pyperclip.copy(ciphertext)
```

الرسالة المشفر سوف يتم طباعتها على الشاشة في السطر 15 ويتم نسخها إلى الذاكرة في السطر 15

البرنامج يقوم بطباعة الرمز ' | ' في نهاية الرسالة المشفرة لكي يتمكن المستخدم من معرفة إذا كان يوجد فراغات في نهاية السلسة المشفرة.

السطر 18 هو آخر سطر في جسم التابع main() وبعد تنفيذه سوف يعود البرنامج لتنفيذ السطر الذي يلي سطر استدعاء هذا التابع.

استدعاء التابع main() هو في السطر 45 في نهاية البرنامج لذلك بعد الانتهاء من تنفيذ آخر سطر من التعليمات الخاصة بهذا التابع سوف ينتهي البرنامج.

البارامترات:

transpositionEncrypt.py

```
21. def encryptMessage(key, message):
```

التابع الذي يقوم بعملية التشفير encryptMessage() يأخذ بارامترين.

البارامترات: هي متغيرات تحوي على قيم يتم تمريرها إلى التابع عند استدعاءه.

البارامترات يتم حذفها بشكل أتوماتيكي بعد تنفيذ التابع.

عندما يتم استدعاء التابع encryptMessage() في السطر 10 ، يتم تمرير قيمتين له (القيمتان الموجودتان في المتغيران myKey and myMessage)

هذه القيم تسند إلى البارامترات key and message (التي يمكن أن تراها في السطر 21)

البارامتر: هو اسم المتغير الموجود بين القوسين ( ) الخاصين بالتابع.

البايثون سوف يظهر رسالة خطأ إذا قمت بتمرير قيم أكثر أو أقل من عدد البارامترات التي يملكها التابع، جرب المثال التالي:

```
>>> len('hello', 'world')
Traceback (most recent call last):
  File "<pyshell#14>", line 1, in <module>
    len('hello', 'world')
TypeError: len() takes exactly one argument (2 given)
>>> len()
Traceback (most recent call last):
  File "<pyshell#15>", line 1, in <module>
    len()
TypeError: len() takes exactly one argument (0 given)
>>>
```

في التابع len() يتم تمرير قيمة واحدة فقط وسوف تحصل على رسالة خطأ في حال تمرير أكثر من قيمة أو عدم تمرير أي قيمة.

تغير البارامترات الموجودة في داخل التابع فقط:

انظر إلى البرنامج التالي الذي يقوم بتعريف ثم استدعاء التابع func()

```
def func(param):
    param = 42
spam = 'Hello'
func(spam)
print(spam)
```

عند تشغيل هذا البرنامج فإن التابع print() الذي تم استدعاءه في السطر الأخير سوف يقوم يثوم بطباعة 'Hello' وليس 42

```
>>>
Hello
>>>
```

عندما يتم استدعاء التابع func() ويتم تمرير المتغير spam له

فإن المتغير spam لن يكون له القيمة 42 وبدل ذلك ستكون له القيمة الموجودة في المتغير param وأي تغير في قيمة param لن تغير قيمة spam

(يوجد استثناء لهذه القاعدة، عندما تقوم بتمرير قيمة لقائمة list أو قاموس dictionary والتي سوف يتم شرحها في الفصل 10)

من الضروري أن تفهم أن المتغير الذي يتم وضعه بين القوسين عند تعريف التابع يسمى بارامیتر وأن القيمة التي يتم تمريرها للتابع عند استدعاءه يتم نسخها إلى البارامیتر، فإذا تغير البارامیتر فإن المتغير الذي يحوي على القيمة التي يتم تمريرها للتابع لن يتغير.

المتغيرات في المجال المحلي والمجال العام:

يمكن أن تكون متفاجئ لماذا قمنا باستخدام البارامترين key and message رغم أننا نملك متغيرين myKey and myMessage في التابع main()

السبب هو أن المتغيران myKey and myMessage في التابع ()main هي متغيرات محلية ولا يمكن استخدامها خارج التابع ()main

في كل مرة يتم فيها استدعاء التابع، يتم خلق مجال محلي local scope

المتغيرات التي يتم خلقها أثناء استدعاء التابع تكون موجودة داخل هذا المجال المحلي.

البارامترات دائماً تكون موجودة في المجال المحلي.

عندما يقوم التابع بإعادة قيمة فإن المجال المحلي يتم تدميره والمتغيرات المحلية يتم نسيانها.

المتغير في المجال المحلي يبقى منفصل عن المتغير في المجال العام (الموجود خارج جسم التابع) حتى ولو كان لهذين المتغيرين نفس الاسم.

المتغيرات التي يتم خلقها خارج التوابع تكون موجودة في المجال العام global scope

عندما ينتهي البرنامج من التنفيذ فيتم تدمير المجال العام وكل المتغيرات الموجودة في البرنامج تُنسى (كل المتغيرات في برنامج الشيفرة العكسية وبرنامج شيفرة قيصر هي متغيرات عامة)

التعليمة global:

إذا كنت تريد لمتغير موجود داخل جسم التابع أن يكون متغير عام قم باستخدام التعليمة global بجانب اسم التابع في أول سطر بعد التعليمة def

التالي هو قواعد المتغيرات العامة والمحلية:

١. المتغيرات خارج التوابع هي دائماً متغيرات عامة
٢. إذا لم يتم تعيين قيمة لمتغير داخل أي تابع فهو متغير عام
٣. إذا كان المتغير داخل التابع فلا يمكن استخدامه في تعليمة عامة ويتم استخدامه فقط كمتغير محلي
٤. إذا تم استخدام التعليمة global مع متغير محلي داخل تابع فيصبح متغير عام

أكتب البرنامج التالي في محرر النصوص واحفظه باسم scope.py واضغط F5 للتنفيذ:

Source code for scope.py

```
1.  spam = 42
2.
3.  def eggs():
4.      spam = 99 # spam in this function is local
5.      print('In eggs():', spam)
6.
7.  def ham():
8.      print('In ham():', spam) # spam in this function is global
9.
10. def bacon():
11.     global spam # spam in this function is global
12.     print('In bacon():', spam)
13.     spam = 0
14.
15. def CRASH():
16.     print(spam) # spam in this function is local
17.     spam = 0
18.
19. print(spam)
20. eggs()
21. print(spam)
22. ham()
23. print(spam)
24. bacon()
25. print(spam)
26. CRASH()
```

نتيجة التنفيذ ستكون كالتالي:

```
42
In eggs():  99
42
In ham() :  42
42
In bacon() :  42
0
Traceback (most recent call last):
  File "C:/Python34/scope.py", line 21, in <module>
    CRASH()
  File "C:/Python34/scope.py", line 12, in CRASH
    print(spam)
UnboundLocalError: local variable 'spam' referenced before assignment
```

عند استخدام المتغير spam في الأسطر 1, 19, 21, 23, 25 فهو خارج كل التوابع فهو متغير عام.

في التابع ()eggs تم إسناد القيمة 99 للمتغير spam لذلك فإن البايثون سوف يعتبر هذا المتغير كمتغير محلي وهو مختلف بشكل عن المتغير العام الذي له نفس الاسم spam

اسناد القيمة 99 إلى المتغير المحلي spam لا تؤثر على قيمة المتغير العام spam (المتغيران مختلفان تماماً ولكن لهما نفس الاسم).

المتغير spam في التابع ()ham لم يتم اسناد أي قيمة له داخل هذا التابع لذلك فهو متغير عام.

المتغير spam في التابع ()bacon يستخدم التعليمة global لذلك فهو متغير عام وعملية اسناد القيمة spam = 0 في السطر 13 سوف تغير قيمة المتغير العام spam.

المتغير spam في التابع ()CRASH تم اسناد قيمة له ولا يتم استخدام التعلية global معه لذلك فهو متغير محلي، ولكن انتبه أنه تم استخدامه في التابع ()print (في السطر 16) قبل أن يتم اسناد قيمة له (السطر 17) وهذا هو سبب ظهور رسالة الخطأ عند استدعاء التابع ()CRASH

```
UnboundLocalError: local variable 'spam' referenced before assignment
```

يمكن أن يكون أمر مشوش في حال استخدام نفس الاسم لمتغيرين محلي وعام لذلك يفضل استخدام أسماء مختلفة.

القائمة List:

```
                                                    transpositionEncrypt.py
22.     # Each string in ciphertext represents a column in the grid.
23.     ciphertext = [''] * key
```

في السطر 23 قمنا باستخدام نوع جديد من البيانات يسمى القائمة list

القائمة يمكن أن تحوي على عدة قيم، وبشكل مماثل للسلسلة النصية التي تبدأ وتنتهي بعلامة التنصيص فإن القائمة تبدأ وتنتهي بالقوسين [ ] ويتم وضع القيم المراد حفظها داخل هذين القوسين ويتم الفصل بين هذه القيم باستخدام الفاصلة ' , '

جرب المثال التالي في الشل التفاعلية:

```
>>> animals = ['cat', 'dog', 'horse']
>>> animals
['cat', 'dog', 'horse']
>>>
```

المتغير animals يحوي على قائمة من القيم وهذه القيم هي ثلاث قيم نصية

كل قيمة داخل القائمة تسمى مادة item

القوائم مفيدة جداً عندما نريد تخزين الكثير من القيم ولا نريد خلق متغير لكل قيمة.

العديد من العمليات يمكن أن تعمل مع القوائم مثل الفهرسة indexing والاقتطاع slicing

دليل الفهرسة يشير إلى المادة داخل القائمة.

جرب المثال التالي:

```
>>> animals = ['cat', 'dog', 'horse']
>>> animals[0]
'cat'
>>> animals[1]
'dog'
>>> animals[2]
'horse'
>>> animals[1:2]
['dog']
>>> animals[0:2]
['cat', 'dog']
>>>
```

تذكر أن دليل الفهرسة يبدأ بالقيمة 0 وليس القيمة 1

يمكن استخدام الحلقة for من أجل المرور على كل القيم الموجودة في القائمة

جرب المثال التالي:

```
>>> for spam in ['cat', 'dog', 'horse']:
        print ('I like ' + spam)


I like cat
I like dog
I like horse
>>>
```

استخدام التابع list() في مجال القائمة:

إذا كنت تريد قائمة من القيم التي تزداد بمقدار معين في كل قيمة يمكنك بناء هذه القائمة باستخدام الحلقة for كما في المثال التالي:

```
>>> myList = []
>>> for i in range(10):
        myList = myList + [i]


>>> myList
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>>
```

يمكن القيام بهذه العملية بشكل مباشر من خلال تمرير التابع range() إلى التابع list()

```
>>> myList = list(range(10))
>>> myList
[0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
>>>
```

التابع list() يمكنه أيضاً تحويل سلسلة نصية إلى قائمة بحيث تكون المواد أو القيم داخل القائمة هي أحرف هذه السلسلة، كما في المثال التالي:

```
>>> myList = list('Hello world!')
>>> myList
['H', 'e', 'l', 'l', 'o', ' ', 'w', 'o', 'r', 'l', 'd', '!']
>>>
```

لن نستخدم التابع list() في مجال القوائم أو مع السلاسل النصية في هذا البرنامج ولكن سيتم ذلك في فصل لاحق في هذا الكتاب.

إعادة تعين المواد في القائمة:

المواد داخل القائمة يمكن أن يتم تعديلها، جرب المثال التالي:

```
>>> animals = ['cat', 'dog', 'horse']
>>> animals
['cat', 'dog', 'horse']
>>> animals[2] = 9999
>>> animals
['cat', 'dog', 9999]
>>>
```

إعادة تعين الأحرف في السلاسل النصية:

يمكننا تعديل المواد في القوائم ولكن لا يمكننا إعادة تعين حرف في سلسلة نصية وفي حال تنفيذ ذلك سوف نحصل على رسالة خطأ

جرب المثال التالي:

```
>>> 'Hello world!'[6] = 'x'
Traceback (most recent call last):
  File "<pyshell#75>", line 1, in <module>
    'Hello world!'[6] = 'x'
TypeError: 'str' object does not support item assignment
>>>
```

من أجل تغير حرف في سلسلة نصية يتم استخدام الاقتطاع

جرب المثال التالي:

```
>>> spam = 'Hello world!'
>>> spam = spam[:6] + 'x' + spam[7:]
>>> spam
'Hello xorld!'
>>>
```

قائمة بمجموعة من القوائم:

القائمة يمكن أن تحوي في داخلها على قوائم أخرى

جرب المثال التالي:

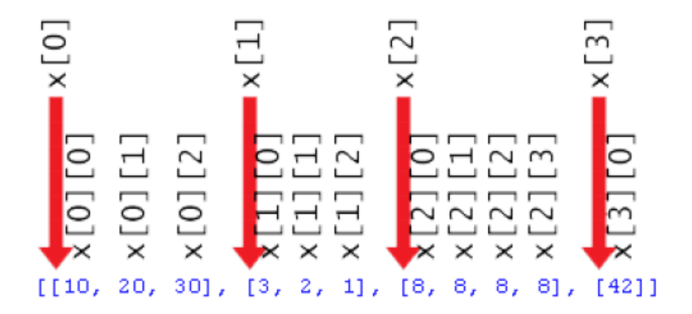
```
>>> spam = [['dog', 'cat'], [1, 2, 3]]
>>> spam[0]
['dog', 'cat']
>>> spam[0][0]
'dog'
>>> spam[0][1]
'cat'
>>> spam[1][0]
1
>>> spam[1][1]
2
>>>
```

كما يمكن استخدام مجموعة أخرى من الفهرسة في حال كانت القيم هس سلاسل نصية

جرب المثال التالي:

```
>>> spam = [['dog', 'cat'], [1, 2, 3]]
>>> spam[0][1][1]
'a'
>>>
```

الشكل التالي يظهر الفهرسة لقائمة اسمها x تحوي على عدد من القوائم في داخلها



استخدام التابع ()len والمعامل in مع القوائم:

نستخدم التابع ()len لمعرفة عدد الأحرف في سلسلة نصية (طول السلسة).

التابع ()len يمكن أن يعمل أيضاً مع القوائم ويعيد عدد صحيح هو عدد المواد الموجودة داخل القائمة.

جرب المثال التالي:

```
>>> animals = ['cat', 'dog', 'horse']
>>> len(animals)
3
>>>
```

المعامل in يستخدم من أجل معرفة فيما إذا كانت سلسلة نصية ما موجودة في داخل سلسلة نصية أخرى.

المعامل in يعمل أيضاً مع القوائم.

جرب المثال التالي:

```
>>> animals = ['cat', 'dog', 'horse']
>>> 'cat' in animals
True
>>> 'dog' in animals
True
>>> 'anteat' in animals
False
>>>
```

كما أن علامات التنصيص الفارغة كانت تعبر عن سلسلة نصية فارغة فإن القوسين [] الفارغين يمثلان قائمة فارغة.

جرب المثال التالي:

```
>>> animals = []
>>> len(animals)
0
>>>
```

جمع وتكرار القوائم باستخدام المعاملات * and + :

بشكل مماثل لاستخدام المعاملات * and + مع السلاسل النصية فإن هذه المعاملات يمكن أن تعمل مع القوائم.

جرب المثال التالي:

```
>>> 'hello ' + 'world'
'hello world'
>>> 'hello ' * 5
'hello hello hello hello hello '
>>>
```

المثال السابق هو للتذكير باستخدام المعاملات * and + مع السلاسل النصية

يمكنك استخدام المعاملات * and + مع القوائم بشكل مشابه للمثال السابق

خوارزمية شيفرة التحويل:

يجب أن نقوم بتحويل الخطوات التي قمنا بها باستخدام الورقة والقلم إلى كود بلغة البايثون

- النص المراد تشفيره 'Common sense is not common. '
- مفتاح التشفير 8

إذا قمت بكتابة أحرف هذا النص داخل الصناديق سوف تحصل على الشكل التالي:

C	o	m	m	o	n	(s)	s
e	n	s	e	(s)	i	s	(s)
n	o	t	(s)	s	o	(s)	c
o	m	m	o	n	.		

دليل الفهرسة لكل حرف في الصناديق (تذكر أن الفهرسة تبدأ من الرقم 0 وليس من الرقم 1)

C 0	o 1	m 2	m 3	o 4	n 5	(s) 6	s 7
e 8	n 9	s 10	e 11	(s) 12	i 13	s 14	(s) 15
n 16	o 17	t 18	(s) 19	s 20	o 21	(s) 22	c 23
o 24	m 25	m 26	o 27	n 28	. 29		

من هذه الصناديق يمكننا أن نرى:

العمود الأول له قيم الفهرسة 0, 8, 16, 24 وهي للأحرف C, e, n, o

العمود الثاني له قيم الفهرسة 1, 9, 17, 25 وهي للأحرف o, n, o, m

ويمكن أن نستنتج بأن العمود رقم n سوف يكون له قيم الفهرسة 0+n, 8+n, 16+n, 24+n

C 0+0=0	o 1+0=1	m 2+0=2	m 3+0=3	o 4+0=4	n 5+0=5	(s) 6+0=6	s 7+0=7
e 0+8=8	n 1+8=9	s 2+8=10	e 3+8=11	(s) 4+8=12	i 5+8=13	s 6+8=14	(s) 7+8=15
n 0+16=16	o 1+16=17	t 2+16=18	(s) 3+16=19	s 4+16=20	o 5+16=21	(s) 6+16=22	c 7+16=23
o 0+24=24	m 1+24=25	m 2+24=26	o 3+24=27	n 4+24=28	. 5+24=29		

لماذا اخترنا الأرقام 0, 8, 16, 24 ؟

هذه الأرقام حصلنا عليها من إضافة قيمة المفتاح (الرقم 8 في هذا المثال) ابتداءً من الرقم 0

0+8=8, 8+8=16, 16+8=24

24+8=32 ولكننا لم نستخدم هذه القيمة لأنها أكبر من عدد أحرف الرسالة ولهذا السبب توقفنا عند الرقم 24

إذا تخيلت قائمة تحوي على ثمانية سلاسل نصية بحيث أن كل سلسلة مكونة من الأحرف الموجودة في كل عمود فإن قيمة هذه القائمة ستكون كالتالي:

```
['Ceno', 'onom', 'mstm', 'me o', 'o sn', 'nio.', ' s ', 's c']
```

بهذه الطريقة يمكننا محاكات الصناديق بكود بلغة بايثون

سنقوم بخلق قائمة فارغة والتي تحوي على عدد من السلاسل النصية الفارغة مساوي لقيمة مفتاح التشفير لأن كل سلسلة نصية سوف تمثل عمود

(في هذا المثال القائمة سوف تحوي على 8 سلاسل نصية لأن مفتاح التشفير هو 8)

```
transpositionEncrypt.py
22.     # Each string in ciphertext represents a column in the grid.
23.     ciphertext = [''] * key
```

المتغير ciphertext هو قائمة تحوي على عدد من القيمة النصية

كل سلسلة نصية في هذه القائمة تمثل عمود، حيث أن :

ciphertext[0] هو العمود الأول من اليسار

ciphertext[1] هو العمود الثاني وهكذا

السلاسل النصية سوف تحوي على كل الأحرف الموجودة داخل كل عمود

0	1	2	3	4	5	6	7
C	o	m	m	o	n	(s)	s
e	n	s	e	(s)	i	s	(s)
n	o	t	(s)	s	o	(s)	c
o	m	m	o	n	.		

المتغير ciphertext الخاص بهذه الشبكة سيكون كالتالي:

```
>>> ciphertext = ['Ceno', 'onom', 'mstm', 'me o', 'nio.', ' s ', 's c']
>>> ciphertext[0]
'Ceno'
>>> ciphertext[1]
'onom'
>>>
```

أول خطوة لخلق هذه القائمة هي خلق عدة سلاسل نصية فارغة بعدد الأعمدة (قيمة مفتاح التشفير)، يمكننا استخدام عملية التكرار من أجل تكرار سلسلة نصية فارغة عدد من المرات بحسب قيمة مفتاح التشفير.

وهذه ما يتم حسابه في السطر 23

transpositionEncrypt.py

```
25.     # Loop through each column in ciphertext.
26.     for col in range(key):
27.         pointer = col
```

الخطوة التالية هي إضافة نص إلى كل سلسلة نصية في القائمة ciphertext

الحلقة for في السطر 26 تقوم بالدوران على كل عمود حيث أن المتغير col سوف يملك قيم تستخدم في الفهرسة في ciphertext

المتغير col سوف يبدأ بالقيمة 0 من أجل أول دورة للحلقة for ثم القيمة 1 من أجل ثاني دورة للحقلة for وهكذا.

وبهذه الطريقة فإن ciphertext[col] سوف تمثل السلسلة النصية الموجودة في العمود ذو الرقم col

أما المتغير pointer فسوف يستخدم في الفهرسة للقيمة النصية المحفوظة في المتغير message ، وفي كل دورة من حلقة for فإن قيمة pointer سوف تبدأ بنفس قيمة col (كما في السطر 27)

معاملات الزيادة:

من أجل اسناد قيمة جديدة لمتغير بالاعتماد على قيمة المتغير الحالية نقوم باستخدام معاملات الزيادة كما في المثال التالي:

```
>>> spam = 40
>>> spam = spam + 2
>>> print(spam)
42
>>>
```

يمكن القيام بهذه العملية من خلال استخدام المعامل =+

جرب المثال التالي:

```
>>> spam = 40
>>> spam += 2
>>> print(spam)
42
>>> spam = 'Hello'
>>> spam += ' world!'
>>> print(spam)
Hello world!
>>>
>>> spam = ['dog']
>>> spam += ['cat']
>>> print(spam)
['dog', 'cat']
>>>
```

التعليمة spam += 2 تعطي نفس نتيجة التعليمة spam = spam + 2 ولكن التعليمة الأولى هي مختصرة.

المعامل =+ يتم استخدامه لإضافة أعداد صحيحة مع بعضها أو لإضافة سلاسل نصية مع بعضها أو حتى لإضافة مواد أو قيم للقوائم.

الجدول التالي يظهر بعض المعاملات والعمليات المكافئة لها:

المعامل	العملية المكافئة
Spam += 2	Spam = spam + 2
Spam -= 2	Spam = spam – 2
Spam *= 2	Spam = spam * 2
Spam /= 2	Spam = spam /2

العودة إلى كود البرنامج:

```
transpositionEncrypt.py
29.         # Keep looping until pointer goes past the length of the message.
30.         while pointer < len(message):
31.             # Place the character at pointer in message at the end of the
32.             # current column in the ciphertext list.
33.             ciphertext[col] += message[pointer]
34.
35.             # move pointer over
36.             pointer += key
```

في داخل الحلقة for التي تبدأ في السطر 26 قمنا بتعريف حلقة while والتي تبدأ في السطر 30 ، من أجل كل عمود نريد المرور على المتغير message ونلتقط كل حرف دليله هو من مضاعفات قيمة مفتاح التشفير.

في السطر 27 ومن أجل أول دورة للحلقة for كانت قيمة pointer هي 0

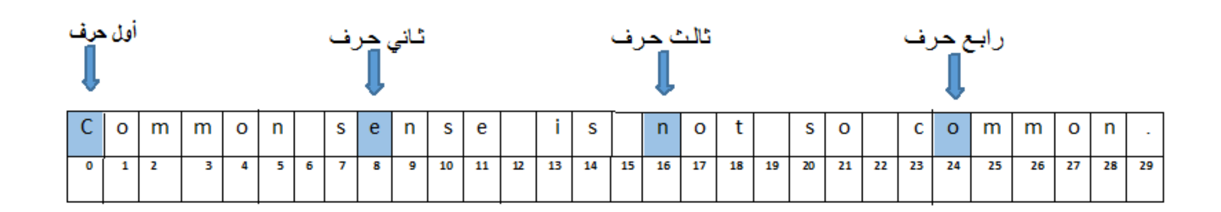
طالما أن قيمة pointer هي أصغر من طول الرسالة، نريد أن يتم إضافة الحرف message[pointer] (الحرف الذي دليل الفهرسة الخاص به هو pointer وهو في هذه الحالة الحرف الأول من الرسالة لأن دليل الفهرسة هو 0) إلى نهاية السلسلة النصية ذات الرقم col داخل القائمة ciphertext

ثم نقوم بإضافة 8 (قيمة مفتاح التشفير) إلى قيمة pointer في كل مرة تتكرر فيها الحلقة (السطر 36)

- في أول دورة للحلقة حصلنا على الحرف message[0]

- في ثاني دورة للحلقة حصلنا على الحرف message[8]
- في ثالث دورة للحلقة حصلنا على الحرف message[16]
- في رابع دورة للحلقة حصلنا على الحرف message[24]

ويتم إضافة هذه الأحرف إلى نهاية السلسلة النصية ciphertext[col] (بما أن col = 0 من أجل أول دورة للحلقة هذا يعني ciphertext[0])

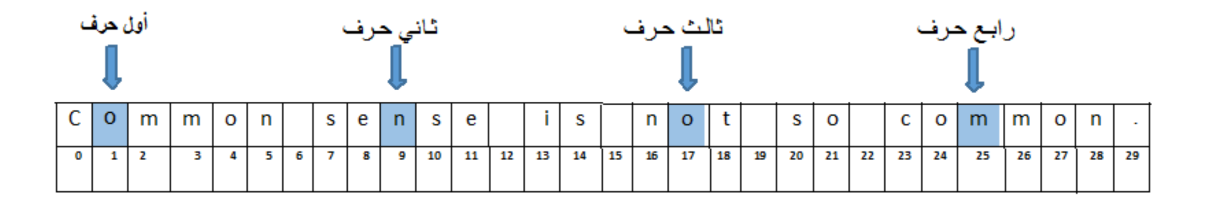


الشكل السابق يظهر كل أحرف الرسالة مع دليل الفهرسة الخاص بكل حرف كما يظهر الأحرف التي تم اختيارها في أو دورة لحلقة for وعند جمع هذه الأحرف مع بعضها سوف نحصل على 'Ceno' وهي السلسلة النصية الأولى في القائمة ciphertest

```
>>> ciphertext = ['Ceno', 'onom', 'mstm', 'me o', 'nio.', ' s ', 's c']
>>> ciphertext[0]
'Ceno'
```

في الدورة الثانية لحلقة for تصبح قيمة المتغير col = 1 (بدلاً من 0)

وتصبح قيمة المتغير pointer = 1 والآن وعند إضافة قيمة المفتاح (8) إلى pointer ضمن حلقة while في السطر 30 فسوف نحصل على قيم الفهرسة 1, 9, 17, 25



message[1], message[9], message[17], message[25]

هذه الأحرف هي مكونات السلسلة النصية الثانية في القائمة ciphertext وهي

ciphertext[1] = 'onom' (وهو العمود الثاني)

عندما تنتهي حلقة for بعد أن تمر على كل العواميد سوف تصبح قيمة القائمة

ciphertext = ['Ceno' , 'onom' , 'me o' , 'o sn' , 'nio.' , ' s ' , 's c']

سوف نستخدم الطريقة النصية join() string method من أجل تحويل هذه السلاسل النصية إلى سلسلة نصية واحدة.

الطريقة النصية join():

الطريقة join() تم استخدامها في السطر 39 من كود البرنامج.

هذه الطريقة تأخذ قائمة من السلاسل النصية وتعيد سلسلة نصية وحيدة هي عبارة عن مجموع هذه السلاسل مع بعضها البعض.

استدعاء هذه الطريقة يجب أن يكون بين السلاسل والقائمة (سوف نستخدم سلسلة نصية فارغة من أجل الاستدعاء)

جرب المثال التالي:

```
>>> eggs = ['dogs', 'cats', 'moose']
>>> ''.join(eggs)
'dogscatsmoose'
>>> ' '.join(eggs)
'dogs cats moose'
>>> 'XYZ'.join(eggs)
'dogsXYZcatsXYZmoose'
>>> ''.join(eggs).upper().join(eggs)
'dogsDOGSCATSMOOSEcatsDOGSCATSMOOSEmoose'
>>>
```

مراحل تنفيذ العبارة الأخيرة ''.join(eggs).upper().join(eggs)

```
''.join(eggs).upper().join(eggs)
''.join(['dogs', 'cats', 'moose']).upper().join(eggs)
'dogscatsmoose'.upper().join(eggs)
'DOGSCATSMOOSE'.join(eggs)
'DOGSCATSMOOSE'.join(['dogs', 'cats', 'moose'])
'dogsDOGSCATSMOOSEcatsDOGSCATSMOOSEmoose'
```

إعادة القيم والتعليمة return:

transpositionEncrypt.py

```
    # Convert the ciphertext list into a single string value and return it.
    return ''.join(ciphertext)
```

استخدامنا للطريقة join() هو استخدام بسيط وليس معقد كما في المثال السابق.

قمنا باستدعاء الطريقة join() على سلسلة نصية فارغة ومررنا لها المتغير ciphertext ليتم جمع السلاسل النصية الموجودة في هذا المتغير مع بعضها البعض (بدون أي فراغات بينها)

التابع أو الطريقة دائماً يتم استدعائهم من أجل حساب قيمة وهذه القيمة تسمى القيمة المعادة (القيمة التي يعيدها التابع أو الطريقة) return value

قمنا بخلق التابع باستخدام التعليمة def كما قمنا باستخدام التعليمة return من أجل أخباره بالقيمة التي يجب عليه إعادتها.

التعليمة return تتبع باسم المتغير المراد إعادة قيمته أو يمكن أن تتبع بعبارة جبرية بدلاً من اسم المتغير وفي هذه الحالة يتم إعادة القيمة بعد حساب العبارة الجبرية.

افتح نافذة جديدة في محرر النصوص واكتب الكود التالي واحفظه باسم addNumbers.py واضغط F5 للتنفيذ:

```
Source code for addNumbers.py
def addNumbers(a, b):
    return a + b

spam = addNumbers(2, 40)
print(spam)
```

عند تنفيذ هذا البرنامج سوف نحصل على النتيجة التالية:

```
42
>>>
```

التابع (addNubers(2, 42 سوف يقوم بحساب القيمة 42 والتعليمة return سوف تقوم بإعادة هذه القيمة

العودة إلى الكود:

```
transpositionEncrypt.py
    # Convert the ciphertext list into a single string value and return it.
    return ''.join(ciphertext)
```

التابع ()encrytMessage يستخدم التعليمة return من أجل إعادة القيمة النصية التي تم تشكيلها من خلال دمج السلاسل النصية الموجودة في القائمة ciphertext

القيمة المعادة هي النص المشفر.

المتغير الخاص __name__:

```
transpositionEncrypt.py
# If transpositionEncrypt.py is run (instead of imported as a module) call
# the main() function.
if __name__ == '__main__':
    main()
```

يمكننا تشغيل برنامجنا في الوحدة module باستخدام اسم المتغير الخاص __name__

عندما يعمل برنامج البايثون هناك متغير خاص __name__ (خطين قبل الاسم وخطين بعده) الذي يتم إسناد القيمة النصية '__main__' له وذلك قبل أول سطر من برنامجك.

في نهاية ملف السكريبت script file وبعد كل تعليمة def نقوم بكتابة بعض التعليمات التي تقوم بفحص فيما إذا كانت قيمة __name__ هي القيمة النصية '__main__' وإذا كان ذلك محققاً نقوم باستدعاء التابع.

التعليمة if في السطر 44 يتم تنفيذها كأول سطر من الكود عند الضغط على F5 (طبعاً بعد تنفيذ تعليمة import في السطر 4 والتعليمة def في السطرين 6 and 21)

السبب لكتابتنا لهذا الكود بهذه الطريقة لأن البايثون يقوم بإسناد القيمة '__main__' إلى المتغير __name__ عند تشغيل البرنامج أو يقوم بإسناد القيمة 'transpositionEncryp' إلى المتغير __name__ إذا تم استيراد هذا البرنامج من قبل برامج أخرى.

بهذه الطريقة يستطيع البايثون معرفة فيما إذا كان هذه البرنامج بدأ العمل كبرنامج مستقل أو تم استيراده من قبل برامج أخرى كما في الوحدات module

بشكل مماثل للطريقة التي يقوم برنامجنا باستيراد الوحدة pyperclip من أجل أن نتمكن من استدعاء التوابع الموجودة في داخلها فهناك برامج يمكن أن تقوم باستيراد برنامجنا من خلال التعليمة import transpositionEncrypt.py من أجل أن تتمكن من استدعاء التابع encryptMessage()

عندما يتم تنفيذ التعليمة import فإن البايثون يقوم بالنظر إلى الملف (الوحدة) module من خلال اللاحقة ' py. ' في نهاية اسم الملف (وهذه هو سبب أن التعليمة import pyperclip تقوم باستيراد الملف pyperclip.py)

عندما يقوم البايثون باستيراد ملف فسوف يتم اسناد اسم هذا الملف إلى المتغير __name__ وثم يبدأ البرنامج بالعمل.

عندما يتم استيراد برنامجنا transpoitionEncrypt.py نريد لكل تعليمات def أن تعمل (من أجل تعريف التابع ()encryptMessage الذي نريد استدعاءه) ولكننا لا نريد استدعاء التابع ()main لأنه يقوم بتنفيذ كود التشفير للسلسلة النصية

'Common sense is not so common.' باستخدام المفتاح 8

وهذه هو سبب وضعنا الكود داخل تابع (التابع ()main) ثم قمنا بإضافة أخر سطرين في البرنامج من أجل استدعاء هذا التابع، عندما نقوم بهذه الأمور فإن برنامجنا يمكن أن يعمل كبرنامج مستقل ويمكن أن يعمل أيضاً كوحدة module لبرنامج أخر.

حجم المفتاح وطول الرسالة:

لاحظ ماذا يحدث عندما يكون طول الرسالة أقل من ضعفي المفتاح

C	o	m	m	o	n	(s)	s	e	n	s	e	(s)	i	s	(s)	n	o	t	(s)	s	o	(s)	c	o
m	m	o	n	.																				

عند استخدام المفتاح 25 لتشفير الرسالة 'Common sense is not so common.' فسوف نحصل على الرسالة المشفرة 'Cmommomno.n sense is not so co' يوجد جزء من الرسالة لم يتم تشفيره !

هذا يحدث عندما يكون حجم المفتاح أكبر من ضعفي طول الرسالة وفي هذه الحالة سوف يوجد على الأقل حرف وحيد في العامود وسوف يتم عرضه في الرسالة المشفرة كما هو في الرسالة الأصلية قبل عملية التشفير.

وبسبب ذلك فإن المفتاح في شيفرة التحويل يجب أن يكون أقل من نصف طول الرسالة المراد تشفيرها.

كلما كانت الرسالة أطول يوجد عدد أكبر من مفاتيح التشفير الممكنة لتشفيرها باستخدام شيفرة التحويل.

الخلاصة:

في هذا الفصل تعلمنا المزيد من أساسيات البرمجة بالإضافة إلى كتابة برنامج لشيفرة التحويل والتي تعتبر أكثر أماناً من شيفرة قيصر.

تعرفنا على التوابع والقوائم ومعاملات الزيادة، تذكر دائماً من أجل أن تفهم أي سطر من كود البرنامج فقط قم بحسابه خطوط بخطوة بنفس الطريقة التي يقوم بها البايثون.

أصبح بإمكاننا ترتيب الكود الخاص بنا ضمن مجموعات تسمى التوابع functions والتي يتم خلقها باستخدام التعليمة def والقيم التي تكون بين أقواس التابع عند تعريفه تسمى البارامترات.

البارامترات هي متغيرات محلية وهي مختلفة عن المتغيرات الموجودة خارج جسم التابع والتي تسمى متغيرات عامة.

تذكر دائماً أن المتغيرات المحلية مختلفة عن المتغيرات العامة حتى ولو كان لها نفس الاسم

القوائم هي نوع من المتغيرات والتي يمكن أن يخزن فيها عدة قيم أخرى ويمكن أن يتم تخزين قائمة ضمن قائمة أخرى.

العديد من العمليات التي يمكن القيام بها مع السلاسل النصية (الفهرسة، التقطيع، التابع ()len) يمكن أيضاً استخدامها مع القوائم.

يمكن استخدام معاملات الزيادة من أجل اختصار التعليمات.

الطريقة النصية ()join تقوم بجمع عدة سلاسل نصية موجودة ضمن قائمة وتعيد سلسلة نصية وحيدة.

إذا لم تفهم مبادئ البرمجة في هذه الفصل بشكل جيد رجاءً قم بالعودة إلى بداية الفصل وأعد القراءة مرة ثانية.

في الفصل القادم سوف نتحدث عن فك التشفير في شيفرة التحويل.



فك التشفير في شيفرة التحويل

محتوى هذا الفصل:

- عملية فك التشفير في شيفرة التحويل
- التوابع math.ceil(), math.floor(), and round()
- المعاملات المنطقية (البوليانية) and , or
- جداول الحقيقة

"When stopping a terrorist attack or seeking to recover a kidnapped child, encountering encryption may mean the difference between success and catastrophic failures."

Attorney General Janet Reno, September 1999

بشكل مختلف عن شيفرة قيصر فإن عملية فك التشفير في شيفرة التحويل هو عملية صعبة

في هذا الفصل سوف نقوم بكتابة البرنامج transpositionDecrypt.py والذي يقوم بعملية فك التشفير

عملية فك التشفير في شيفرة التحويل على الورق:

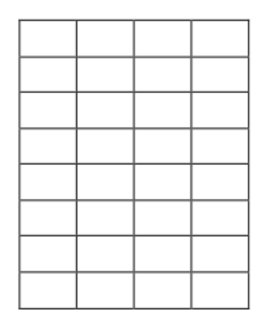
لنفترض أننا قمنا بإرسال النص المشفر 'Cenoonommstmme oo snnio. s s c' إلى صديق والذي يعرف مفتاح التشفير السري المستخدم في عملية التشفير وهو الرقم 8

أول خطوة يقوم بها هذا الصديق من أجل فك تشفير هذه الرسالة هو حساب عدد الصناديق التي يجب أن يقوم برسمها

من أجل القيام بهذه المهمة يجب أن يقوم بتقسيم طول الرسالة المشفرة على قيمة المفتاح وتقريب النتيجة للعدد الصحيح التالي

في هذا المثال طول الرسالة المشفرة هو 30 حرف (نفس طول النص الصريح) ومفتاح التشفير هو الرقم 8 وبالتالي 3.75 = 30/8

يتم تقريب النتيجة إلى العدد الصحيح التالي وهو 4 ، هذا يعني أنه يجب رسم شبكة من الصناديق تحوي على أربع عواميد (الرقم الذي قمنا بحسابه) وثمانية أسطر (قيمة المفتاح)



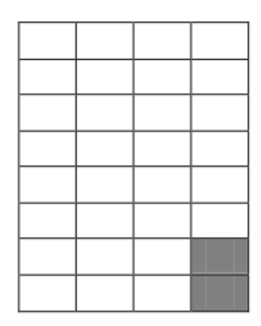
انتبه في حالة كانت نتيجة التقسيم هو عدد صحيح فنأخذ العدد الناتج كما هو دون القيام بأي عملية تقريب

مثلاً 6.0 = 30/5 في هذه الحالة نأخذ الرقم 6 ولا نقوم بأي عملية تقريب

الخطوة التالية هي حساب عدد الصناديق في أقصى اليمين والتي يجب تظليلها

عدد الصناديق الكلي هو 32 وطول الرسالة المشفرة هو 30 وبالتالي 2 = 30 – 32

يجب تظليل صندوقين في الجانب الأيمن



ثم تبدأ عملية ملئ الصناديق بالأحرف، كل حرف في صندوق حيث يتم البدء من الزاوية العلوية اليسرى ثم باتجاه اليمين كما في عملية التشفير

الرسالة المشفرة هي 'Cenoonommstmme oo snnio. s s c' لذلك يكون :

محتوى أول سطر هو 'Ceno'

محتوى ثاني سطر هو 'onom' وهكذا

الفراغات يتم تمثيلها ب (s)

C	e	n	o
o	n	o	m
m	s	t	m
m	e	(s)	o
o	(s)	s	n
n	i	o	.
(s)	s	(s)	
s	(s)	c	

صديقنا الذي استلم هذه الرسالة المشفرة سوف يرها إذا قام بقراءة النص بشكل عامودي من كل عامود وسوف يحصل على النص الصريح لهذه الرسالة وهو

'Common sense is not so common.'

خطوات عملية فك التشفير:

١. حساب عدد الأعمدة ويتم ذلك من خلال تقسيم طول الرسالة على المفتاح وتقريب النتيجة إلى العدد الصحيح التالي

٢. رسم شبكة من الصناديق بحيث يكون عدد الأعمدة هو العدد الذي قمنا بحسابه في الخطوة الأولى وعدد الأسطر هو نفس قيمة المفتاح
٣. حساب عدد الصناديق التي يجب تظليلها من خلال طرح عدد أحرف الرسالة من عدد الصناديق الكلي
٤. تظليل الصناديق التي تم حساب عددها في الخطوة الثالثة في الزاوية اليمنية السفلى
٥. ملئ الصناديق بأحرف الرسالة المشفرة بدأً من أول سطر ومن اليسار إلى اليمين ويتم تجاهل الصناديق المظللة
٦. الحصول على النص الصريح من خلال البدء بالقراءة من العمود الأيسر ومن الأعلى إلى الأسفل ثم الانتقال إلى العمود التالي

في حال استخدام مفتاح مختلف سوف يتم رسم عدد خاطئ من العواميد وحتى لو اتبعت باقي الخطوات بشكل صحيح فلن نحصل على النص الصريح إلا إذا استخدمت قيمة المفتاح الصحيحة

برنامج فك التشفير لشيفرة التحويل:

افتح نافذة جديدة في محرر النصوص وأكتب الكود التالي ثم احفظه باسم transpositionDecrypt.py

كود برنامج فك التشفير لشيفرة التحويل:

افتح نافذة جديدة من محرر النصوص File > New File واكتب الكود التالي ثم قم بحفظه باسم transpositionDecrypt.py واضغط F5 من أجل التنفيذ

يجب أن تضع الملف (الوحدة) pyperclip.py في نفس المجلد الذي سوف تحفظه به ملف برنامج transpositionDecrypt.py

يمكنك الحصول على pyperclip.py من http://invpy.com/pyperclip.py

Source code for transpositionDecrypt.py

```python
# Transposition Cipher Decryption
# http://inventwithpython.com/hacking (BSD Licensed)

import math, pyperclip

def main():
    myMessage = 'Cenoonommstmme oo snnio. s s c'
    myKey = 8

    plaintext = decryptMessage(myKey, myMessage)

    # Print with a | (called "pipe" character) after it in case
    # there are spaces at the end of the decrypted message.
    print(plaintext + '|')

    pyperclip.copy(plaintext)


def decryptMessage(key, message):
    # The transposition decrypt function will simulate the "columns" and
    # "rows" of the grid that the plaintext is written on by using a list
    # of strings. First, we need to calculate a few values.

    # The number of "columns" in our transposition grid:
    numOfColumns = math.ceil(len(message) / key)
    # The number of "rows" in our grid will need:
    numOfRows = key
    # The number of "shaded boxes" in the last "column" of the grid:
    numOfShadedBoxes = (numOfColumns * numOfRows) - len(message)

    # Each string in plaintext represents a column in the grid.
    plaintext = [''] * numOfColumns

    # The col and row variables point to where in the grid the next
    # character in the encrypted message will go.
    col = 0
    row = 0

    for symbol in message:
        plaintext[col] += symbol
        col += 1 # point to next column

        # If there are no more columns OR we're at a shaded box, go back to
        # the first column and the next row.
        if (col == numOfColumns) or (col == numOfColumns - 1 and row >= numOfRows - numOfShadedBoxes):
            col = 0
            row += 1

    return ''.join(plaintext)


# If transpositionDecrypt.py is run (instead of imported as a module) call
# the main() function.
if __name__ == '__main__':
    main()
```

نتيجة تنفيذ هذا البرنامج هي:

```
Common sense is not so common.|
>>>
```

إذا كنت تريد فك تشفير رسالة مختلفة أو القيام بعملية فك التشفير باستخدام قيمة مفتاح مختلفة يمكنك تغير قيم المتغيران myMessage, myKey في السطرين 5 and 6

كيف يعمل هذا البرنامج:

transpositionDecrypt.py

```
1. # Transposition Cipher Decryption
2. # http://inventwithpython.com/hacking (BSD Licensed)
3.
4. import math, pyperclip
5.
6. def main():
7.     myMessage = 'Cenoonommstmme oo snnio. s s c'
8.     myKey = 8
9.
10.     plaintext = decryptMessage(myKey, myMessage)
11.
12.     # Print with a | (called "pipe" character) after it in case
13.     # there are spaces at the end of the decrypted message.
14.     print(plaintext + '|')
15.
16.     pyperclip.copy(plaintext)
```

أول جزء من هذا البرنامج شبيه جداً بالجزء الأول من برنامج transpositionEncrypt.py

الوحدة pyperclip والوحدة math يتم استيرادهم باستخدام التعليمة import

يتم خلق التابع main() والذي يحوي على بارامترين myMessage, myKey ثم يتم استدعاء التابع decryptMessage()

القيمة التي يعيدها هذا التابع هي النص الصريح والتي يتم بعدها طباعتها على الشاشة (بعد الرسالة يتم طباعة الرمز ' | ') ويتم نسخ هذا النص إلى الذاكرة

```
transpositionDecrypt.py
19. def decryptMessage(key, message):
```

انظر إلى خطوات عملية فك التشفير في بداية هذا الفصل

مثلاً إذا كنت تريد فك تشفير الرسالة 'Cenoonommstmme oo snnio. s s c' (المكونة من 30 حرف) باستخدام المفتاح 8 فيجب عليك رسم وملئ الصناديق كما في الشكل التالي:

C	e	n	o
o	n	o	m
m	s	t	m
m	e	(s)	o
o	(s)	s	n
n	i	o	.
(s)	s	(s)	
s	(s)	c	

التابع ()decryptMessage يقوم بخطوات عملية فك التشفير

التوابع ()math.ceil, math.floor(), round:

عندما تقوم بتقسيم الأرقام باستخدام معامل القسمة ' / ' فإن النتيجة التي سيتم إعادتها هي عدد حقيقي ذو فاصلة عشرية floating point number حتى ولو كانت نتيجة القسم عدد بدون باقي.

مثلاً: 21 / 7 = 3.0 وليس 3

```
>>> 21 / 7
3.0
```

وفي حال كانت نتيجة القسمة تحوي على باقي ستظهر النتيجة رقم مع فاصلة عشرية كما في المثال التالي:

22 / 5 = 4.4

```
>>> 22 / 5
4.4
```

إذا حصلت على نتيجة 4 = 22 / 5 بدلاً من 4.4 هذا يعني أنك تستخدم Python 2 وليس Python 3 اذهب إلى http://python.org وقم بتحميل وتنصيب Python 3

إذا أردت تقريب أو تدوير الرقم إلى أقري عدد صحيح يمكنك استخدام التابع round()

جرب المثال التالي:

```
>>> round(4.2)
4
>>> round(4.5)
4
>>> round(4.9)
5
>>> round(5.0)
5
>>> round(22 / 5)
4
>>> .
```

إذا كنت تريد التقريب إلى العدد التالي يمكنك استخدام التابع math.ceil() والذي هو اختصار للكلمة 'ceiling' والتي تعني سقف أو حد أعلى

إذا كنت تريد التقريب إلى العدد الأقل يمكنك استخدام التابع math.floor()

هذه التوابع موجودة في الوحدة math module لذلك يجب أن تقوم باستيراد هذه الوحدة لكي تستطيع استدعاء هذه التوابع.

جرب المثال التالي:

```
>>> import math
>>> math.floor(4.0)
4
>>> math.floor(4.2)
4
>>> math.floor(4.9)
4
>>> math.ceil(4.0)
4
>>> math.ceil(4.2)
5
>>> math.ceil(4.9)
5
>>>
```

التابع ()math.ceil يقوم بالخطوة الأولى من خطوات عملية فك التشفير لشيفرة التحويل.

```
                                                              transpositionDecrypt.py
19. def decryptMessage(key, message):
20.     # The transposition decrypt function will simulate the "columns" and
21.     # "rows" of the grid that the plaintext is written on by using a list
22.     # of strings. First, we need to calculate a few values.
23.
24.     # The number of "columns" in our transposition grid:
25.     numOfColumns = math.ceil(len(message) / key)
26.     # The number of "rows" in our grid will need:
27.     numOfRows = key
28.     # The number of "shaded boxes" in the last "column" of the grid:
29.     numOfShadedBoxes = (numOfColumns * numOfRows) - len(message)
```

في السطر 25 يتم حساب عدد الأعمدة (الخطوة الأولى في عملية فك التشفير) من خلال تقسم طول الرسالة (len(message على قيمة المفتاح ثم يتم تمرير النتيجة إلى التابع ()math.ceil ويتم حفظ النتيجة النهائية في المتغير numOfColumns .

في السطر 27 يتم حساب عدد الأعمدة (الخطوة الثانية في عملية فك التشفير) وهو عدد صحيح مساوي لقيمة المفتاح ويتم حفظ هذه القيمة في المتغير numOfRows .

في السطر 29 يتم حساب عدد الصناديق المظللة (الخطوة الثالثة في عملية فك التشفير) وهو عدد الصناديق الكلي (عدد الأسطر ضرب عدد الأعمدة) منقوصاً منه طول الرسالة.

```
                                                              transpositionDecrypt.py
31.     # Each string in plaintext represents a column in the grid.
32.     plaintext = [''] * numOfColumns
```

المتغير plaintext هو قائمة تحوي على عدد من السلاسل النصية وهذه السلاسل تبدأ فارغة ثم يتم ملئ كل سلسلة بأحرف كل عمود

استخدام عملية الضرب سوف يؤدي لتكرار السلسلة النصية الفراغة بحسب عدد الأعمدة.

تذكر أنه عند استدعاء أي تابع سوف يتم خلق مجال محلي خاص به والمتغير plaintext موجود بداخل التابع ()decryptMessage وهذا يعني أن المتغير plaintext هو متغير محلي وهو مختلف عن المتغير الذي له نفس الاسم ()plaintext والموجود ضمن التابع ()main

شبكة الصناديق في مثالنا لها الشكل التالي:

C	e	n	o
o	n	o	m
m	s	t	m
m	e	(s)	o
o	(s)	s	n
n	i	o	.
(s)	s	(s)	
s	(s)	c	

المتغير plaintext هو قائمة تحوي على مجموعة من السلاسل النصية وكل سلسلة نصية مخصصة لعمود واحد وستكون قيمة المتغير plaintext كالتالي:

```
>>> plaintext = ['Common s' , 'ense is ' , 'not so c' , 'ommon.']
>>> plaintext[0]
'Common s'
>>>
```

يمكننا استدعاء الطريقة join() لتقوم بجمع هذه السلاسل النصية مع بعضها البعض لنحصل على سلسلة نصية واحدة وهي 'Common sense is not so common.'

```
transpositionDecrypt.py
34.     # The col and row variables point to where in the grid the next
35.     # character in the encrypted message will go.
36.     col = 0
37.     row = 0
38.
39.     for symbol in message:
```

المتغيران col and row سوف يحددان السطر والعمود الذي سيكون فيه الحرف التالي

تبدأ قيم المتغيران بالرقم 0 وفي السطر 39 نستخدم حلقة for من أجل المرور على أحرف الرسالة الموجودة في المتغير message وبداخل هذه الحلقة سوف يتم كتابة كود يقوم بتعديل قيم المتغيران col and row وسوف نستمر بهذه العملية إلى أن نحصل على السلسلة النصية الصحيحة التي يتم حفظها في المتغير من نوع قائمة plaintext

```
transpositionDecrypt.py
40.         plaintext[col] += symbol
41.         col += 1 # point to next column
```

أول خطوة في داخل حلقة for هي إضافة الرمز symbol إلى السلسلة النصية ذات قيمة الفهرسة col داخل القائمة plaintext

ثم نقوم بزيادة قيمة col بمقدار واحد وذلك في السطر 41 وبهذه الطريقة سوف يتم إضافة باقي الرموز من خلال تكرار الحلقة

المعاملات المنطقية and , or:

المعاملات المنطقية and , or تساعدنا في الشروط المعقدة الخاصة بتعليمات if and while

المعامل and يوضع بين عبارتين جبريتين ويعيد القيمة True إذا كانت كلتا العبارتين الجبريتين محققتين.

المعامل or يوضع بين عبارتين جبريتين ويعيد القيمة True إذا كانت إحدى العبارتين محققة.

جرب المثال التالي في الشيل التفاعلية:

```
>>> 10 > 5 and 2 < 4
True
>>> 10 > 5 and 4 != 4
False
>>>
```

العملية الأولى تعيد القيمة True لأن كلا العبارتين بجانب المعامل and محققتين

True and True = True

العملية الثانية تعيد القيمة False لأن العبارة الولى محققة والثانية غير محققة

True and False = False

جرب المثال التالي:

```
>>> 10 > 5 or 4 != 4
True
>>> 10 < 5 or 4 != 4
False
>>>
```

المعامل or يعيد القيمة True إذا كانت إحدى القيم على جانبه على الأقل هي True ويعيد False إذا كانت كلتا القيميتين على جانبيه هي False.

المعامل المنطقي not يقوم بعكس القيمة المنطقية

not True = False

not False = True

جرب المثال التالي:

```
>>> not 10 > 5
False
>>> not 10 < 5
True
>>> not False
True
>>> not not False
False
>>> not not not not not False
True
>>>
```

جداول الحقيقة:

إذا نسيت كيفية عمل المعاملات المنطقية يمكن النظر إلى هه الجداول والتي تسمى جداول الحقيقة:

جدول الحقيقة للمعامل and

A	and	B	النتيجة
True	and	True	**True**
True	and	False	**False**
False	and	True	**False**
False	and	False	**False**

جدول الحقيقة للمعامل or

A	or	B	النتيجة
True	or	True	**True**
True	or	False	**True**
False	or	True	**True**
False	or	False	**False**

جدول الحقيقة للمعامل not

not A	النتيجة
not True	False
not False	True

المعامل and يمكن أن يستخدم من أجل اختصار تعليمات if

جرب المثال التالي:

```
>>> if 10 > 5:
        if 2 < 4:
                print('Hello!')


Hello!
>>>
>>> if 10 > 5 and 2 < 4:
        print('Hello!')


Hello!
>>>
```

كما يمكننا أيضاً استخدام المعامل or بدلاً من التعليمات if , elif

جرب المثال التالي:

```
>>> if 4 != 4:
	print('Hello!')
elif 10 > 5:
	print('Hello!')

	
Hello!
>>> 
>>> if 4 != 4 or 10 > 5:
	print('Hello!')

	
Hello!
>>> 
```

ترتيب العمليات في المعاملات المنطقية:

كما في العمليات الرياضية فإن المعاملات المنطقية and, or, not لها ترتيب لتنفيذ العمليات

العملية ذات الأولوية العليا هي not ثم and ثم or

جرب المثال التالي:

```
>>> not False and True
True
>>> not (False and True)
True
>>> 
```

العودة إلى البرنامج:

```
transpositionDecrypt.py
        # If there are no more columns OR we're at a shaded box, go back to
        # the first column and the next row.
        if (col == numOfColumns) or (col == numOfColumns - 1 and row >= numOfRows - numOfShadedBoxes):
            col = 0
            row += 1
```

يوجد حالتين نريد عندهما ضبط قيمة col = 0

الحالة الأولى: إذا ازدادت قيمة col وأصبحت أكبر من آخر دليل فهرسة للقائمة plaintext وفي هذه الحالة فإن القيمة في المتغير col سوف تساوي numOfColumns (تذكر أن آخر دليل فهرسة للقائمة plaintext هو numOfColumns)

الحالة الثانية: إذا كان col له قيمة آخر دليل فهرسة و المتغير row كان يشير إلى الصناديق المظللة في آخر عمود

عندها فإن كامل شبكة فك التشفير مع دليل الفهرسة لكل حرف ستكون بالشكل التالي:

	0	1	2	3
0	C 0	e 1	n 2	o 3
1	o 4	n 5	o 6	m 7
2	m 8	s 9	t 10	m 11
3	m 12	e 13	(s) 14	o 15
4	o 16	(s) 17	s 18	n 19
5	n 20	i 21	o 22	. 23
6	(s) 24	s 25	(s) 26	
7	s 27	(s) 28	c 29	

كما ترى فإن الصناديق المظللة موجودة في آخر عمود (عندما يكون دليل الفهرسة هو numOfColumns -1) وفي السطرين 6 and 7

قمنا باستخدام العبارة الجبرية row >= numOfRows – numOfShadedBoxes

من أجل حساب دليل الفهرسة للأسطر التي تحوي على الصناديق المظللة

فإذا كانت هذه العبارة محققة True وكان col = numOfColumns -1 عندها سوف نقوم بإعادة ضبط قيمة col لتعود إلى القيمة 0

الشرط الخاص بتعليمة if هو

(col == numOfColumns) or (col == numOfColumns -1 and row >= (numOfRows – numOfShadedBoxes)

إذا كان هذا الشرط محقق سوف يقوم البرنامج بتنفيذ كتلة التعليمات التالية وهي تغير قيمة col ليعود لأول عمود من خلال إعادة تعيين قيمته لتصبح مساوية للقيمة صفر وزيادة القيمة الموجودة في المتغير row بمقدار واحد

```
transpositionDecrypt.py
    return ''.join(plaintext)
```

عندما تنتهي حلقة for (التي بدأت في السطر 39) من الدوران على كل الأحرف الموجودة في المتغير message ستكون قيم السلاسل النصية الموجودة في القائمة plaintext هي أحرف السلسلة النصية بعد عملية فك التشفير (طبعاً إذا تم استخدام مفتاح التشفير الصحيح)

السلاسل النصية الموجودة في القائمة plaintext يتم جمعها مع بعضها باستخدام الطريقة النصية ()join والسلسلة النصية الناتجة هي القيمة التي سوف يعيدها التابع decryptMessage()

في هذا المثال فإن القائمة plaintext ستكون كالتالي:

Plaintext = ['Common s' , 'ense is ' , 'not so c' , 'ommon.']

والتعليمة ' '.join(plaintext) سوف تعيد السلسلة النصية التالية:

'Common sense is not so common.'

```
transpositionDecrypt.py
52. # If transpositionDecrypt.py is run (instead of imported as a module) call
53. # the main() function.
54. if __name__ == '__main__':
55.     main()
```

أول سطر سوف يقوم البرنامج بتنفيذه بعد استيراد الوحدات وتنفيذ التعليمات def هو السطر 54 حيث نقوم بفحص فيما إذا كان هذا البرنامج يعمل بشكل مستقل أو تم استيراده من قبل برنامج أخر من خلال فحص قيمة المتغير الخاص __name__ إذا كانت مساوية للقيمة '__main__' إذا كان هذا محقق عندها يتم تنفيذ التابع main()

الخلاصة:

من أجل برنامج فك التشفير قمنا بوضع معظم الكود داخل التابع ()decryptMessage

برامجمنا يمكنها تشفير وفك تشفير الرسالة '.Common sense is not so common' باستخدام المفتاح ذو القمية 8 ولكن يجب أن نجرب رسائل مختلفة وقيمة مفتاح تشفير مختلفة

ورؤية فيما إذا كانت النتيجة هي نفس الرسالة الأصلية

يمكننا تغير قيمة المتغيرات key , message في كل من البرنامجين transpositionEncrypt.py and transpositionDecrypt.py ومن ثم تشغيل هذين البرنامجين لمعرفة إذا كانا يعملان أو لأ ولكن بدل ذلك سنقوم بكتابة برنامج يقوم بهذه العملية بشكل اتوماتيكي



تشفير وفك تشفير الملفات

محتوى هذا الفصل:

- قراءة وكتابة الملفات
- التابع ()open
- الطريقة ()read
- الطريقة ()close
- الطريقة ()write
- التابع ()os.path.exists
- الطريقة النصية ()statwith
- الطريقة النصية ()title
- الوحدة time والتابع ()time.time

إلى الآن فإن برامجنا تعمل فقط مع رسائل قصيرة نقوم بكتابتها بشكل مباشر في الكود البرمجي للبرنامج كسلسلة نصية

برنامج التشفير في هذا الفصل سوف يستخدم شيفرة التحويل transposition cipher من أجل تشفير وفك تشفير ملفات كاملة والتي يمكن أن تحوي على ملايين الأحرف والكلمات.

ملفات النص الصريح:

البرنامج في هذا الفصل سوف يقوم بتشفير وفك تشفير ملفات تحوي على النص الصريح plain text files وهذا النوع من الملفات يحوي فقط على بيانات نصية وغالباً ما يكون امتداد هذه الملفات هو txt.

الملفات التي تنتج من برامج المعالجة التي تسمح لك بتغير الخط واللون وحجم النص هي ليست ملفات نص صريح.

يمكنك كتابة الملفات النصية في نظام التشغيل ويندوز باستخدام محرر النصوص مثل Notepad أو في نظام التشغيل لينكس باستخدام gedit أو باستخدام أي محرر نصوص أخر

كما يمكنك استخدام IDLE كمحرر نصوص ومن خلاله تقوم بحفظ الملفات بامتداد txt. بدل من استخدام الامتداد py.

يمكنك تحميل بعض الأمثلة لملفات نصية من موقع هذا الكتاب:

- http://invpy.com/devilsdictionary.txt
- http://invpy.com/frankenstein.txt
- http://invpy.com/siddhartha.txt
- http://invpy.com/thetimemachine.txt

بعد تحميل الملف النصي الموجود في الرابط http://invpy.com/frankenstein.txt قم بفتح الملف النصي من خلال برنامج محرر النصوص.

هذه الملف يحوي على أكثر من 78,000 كلمة وطبعاً هذا سوف يأخذ وقت طويل جداً من أجل كتابة هذه الكلمات في برنامج التشفير ولكن إذا كانت هذه الكلمات موجودة في ملف نصي فإن البرنامج يستطيع قراءة هذا الملف والقيام بعملية التشفير بأقل من ثانيتين.

إذا حصلت على رسالة خطأ مشابهه للرسالة التالية

```
“UnicodeDecodeError: 'charmap' codec can't decode byte 0x90 in position 148: character maps to <undefined>”
```

هذا بسبب أنك تقوم بتشغيل برنامج التشفير على ملف نص نصي غير صريح (مشفر)

non-plain text file

الكود البرمجي لبرنامج تشفير الملفات بشيفرة التحويل:

Source Code of the Transpostion File Cipher Program

بشكل مشابه لبرنامج شيفرة التحويل فإن برنامج تشفير الملفات بشيفرة التحويل سوف يقوم باستيراد الملفات transpositionEncrypt.py and transpsitionDecrypt.py لنتمكن من استخدام التوابع encryptMwssage() and decryptMessage() الموجودة في داخلهما.

وبهذه الطريقة لن نكوم بحاجة لإعادة كتابة الكود البرمجي الخاص بهذه التوابع في برنامجنا الجديد.

قم بفتح ملف جديد في محرر النصوص من خلال File > New File وقم بكتابة الكود التالي ثم قم بحفظه باسم transpoitionFileCipher.py واضغط F5 من أجل تشغيل البرنامج.

يجب أن تقوم بتحميل الملف النصي Frankenstein.txt ووضعه في نفس المجلد الذي يحوي على ملف transpostionFileCipher.py

يمكنك تحميل هذا الملف النصي من http://invpy.com/frankenstein.txt.

Source code for transpositionFileCipher.py

```
# Transposition Cipher Encrypt/Decrypt File
# http://inventwithpython.com/hacking (BSD Licensed)

import time, os, sys, transpositionEncrypt, transpositionDecrypt

def main():
    inputFilename = 'frankenstein.txt'
    # BE CAREFUL! If a file with the outputFilename name already exists,
    # this program will overwrite that file.
    outputFilename = 'frankenstein.encrypted.txt'
    myKey = 10
    myMode = 'encrypt' # set to 'encrypt' or 'decrypt'

    # If the input file does not exist, then the program terminates early.
    if not os.path.exists(inputFilename):
        print('The file %s does not exist. Quitting...' % (inputFilename))
        sys.exit()

    # If the output file already exists, give the user a chance to quit.
    if os.path.exists(outputFilename):
        print('This will overwrite the file %s. (C)ontinue or (Q)uit?' % (outputFilename))
        response = input('> ')
        if not response.lower().startswith('c'):
            sys.exit()

    # Read in the message from the input file
    fileObj = open(inputFilename)
    content = fileObj.read()
    fileObj.close()

    print('%sing...' % (myMode.title()))

    # Measure how long the encryption/decryption takes.
    startTime = time.time()
    if myMode == 'encrypt':
        translated = transpositionEncrypt.encryptMessage(myKey, content)
    elif myMode == 'decrypt':
        translated = transpositionDecrypt.decryptMessage(myKey, content)
    totalTime = round(time.time() - startTime, 2)
    print('%sion time: %s seconds' % (myMode.title(), totalTime))

    # Write out the translated message to the output file.
    outputFileObj = open(outputFilename, 'w')
    outputFileObj.write(translated)
    outputFileObj.close()

    print('Done %sing %s (%s characters).' % (myMode, inputFilename, len(content)))
    print('%sed file is %s.' % (myMode.title(), outputFilename))


# If transpositionCipherFile.py is run (instead of imported as a module)
# call the main() function.
if __name__ == '__main__':
    main()
```

في المجلد الذي يحوي على الملفات Frankenstein.txt and transpoitionFileCipher.py سوف يتم خلق ملف جديد بعد تنفيذ البرنامج وله الاسم Frankenstein.rncrypted.txt والذي يحوي على المحتوى المشفر للنص الموجود في الملف Frankenstein.txt

من أجل القيام بعملية فك التشفير قم بغجراء التعديلات التالية على الكود البرمجي (الكلمات المكتوبة بالخط العريض) ثم قم بتشغيل البرنامج مرة ثانية

```
transpositionFileCipher.py
    inputFilename = 'frankenstein.encrypted.txt'
    # BE CAREFUL! If a file with the outputFilename name already exists,
    # this program will overwrite that file.
    outputFilename = 'frankenstein.decrypted.txt'
    myKey = 10
    myMode = 'decrypt' # set to 'encrypt' or 'decrypt'
```

عندما تقوم بتشغيل البرنامج بعد إجراء هذا التعديل سوف يظهر في نفس المجلد ملف باسم Frankenstein.decrypted.txt والذي يحوي على النص الأصلي الموجود في الملف Frankenstein.txt

عندما تقوم بتشغيل برنامج التشفير السابق سوف تحصل على الخرج التالي:

```
Encrypting...
Encryption time: 1.21 seconds
Done encrypting frankenstein.txt (441034 characters).
Encrypted file is frankenstein.encrypted.txt.
```

الملف Frankenstein.encrypted.txt يتم خلقه في نفس المجلد الذي يحوي على ملف برنامج التشفير transpositionFileCipher.py

اذا قمت بفتح هذا الملف باستخدام محرر النصوص سوف ترى النص المشفر لمحتوى الملف النصي Frankenstein.txt

يمكنك الآن إرسال هذا النص المشفر إلى أي شخص عبر الايميل ليقوم هو بعملية فك التشفير وقراءة نص الرسالة.

القراءة من الملفات:

حتى الآن كل دخل نريد أن نعطيه للبرنامج هو دخل يقوم المستخدم بكتابته

برامج بايثون يمكنها فتح وقراءة الملفات بشكل مباشر من القرص الصلب

يوجد ثلاث خطوات من أجل قراءة محتوى أي ملف:

١. فتح الملف
٢. قراءة المحتوى عند طريق اسناده إلى متغير
٣. إغلاق الملف

التابع ()open و file objects:

أول برامتر يتم تمريره للتابع ()open هو اسم الملف المراد فتحه.

إذا كان الملف في نفس المجلد الذي يحوي على برنامج البايثون يكفي فقط كتابة اسم الملف أو يمكنك أيضاً كتابة المسار الذي يحوي على الملف مثل

'c:\\Python32\\frankentein.txt' في نظام التشغيل ويندوز

أو '/usr/foobar/frankentein.txt' في نظام التشغيل لينكس

التابع ()open يعيد قيمة من نوع البيانات "file object" ويوجد عدة طرق من أجل قراءة هذه القيمة أو الكتابة فيها أو إغلاق الملف.

الطريقة ()read الخاصة ب File Object:

الطريقة ()read تقوم بإعادة سلسلة نصية تحوي على كل النص الموجود داخل الملف، لنفترض أن الملف النصي spam.txt يحوي على النص التالي "Hello world!" (يمكنك خلق هذا الملف بنفس) وعند تشغيل الكود التالي في الشيل التفاعلية (هذا الكود يفترض أنك تعمل على نظام التشغيل ويندوز وأنك قمت بخلق ملف نصي باسم spam.txt في المسار c:\

```
>>> fo = open("c:\\spam.txt.txt", 'r')
>>> content = fo.read()
>>> print(content)
```

نتيجة تنفيذ الكود السابق هي:

```
Hello World
>>>
```

إذا كان الملف النصي يحوي على أكثر من سطر، فإن السلسلة النصية التي المُعادة من الطريقة read() سوف تحوي على \n للدلالة على نهاية كل سطر

عندما تحاول طباعه سلسلة نصية بأكثر من سطر يمكنك استخدام \n كما في المثال التالي:

```
>>> print('Hello\nworld')
Hello
world
>>>
```

إذا حصلت على رسالة الخطأ التالية:

```
"IOError: [Errno 2] No such file or directory"
```

قم بالتأكد من اسم الملف أو المسار الذي يحوي على هذا الملف.

الطريقة close() الخاصة ب File Object:

بعد قراءة محتوى الملف وإسناده إلى متغير، يمكنك اعلام البايثون بأنك قد انتهيت من هذا الملف من خلال استدعاء الطريقة close()

```
>>> fo.close()
>>>
```

البايثون يقوم بشكل اتوماتيكي بإغلاق أي ملف مفتوح عند انتهاء البرنامج الذي يستخدم هذا الملف وعندما تريد القراءة من هذا الملف مرة ثانية يجب أن تقوم باغلاق الملف ومن ثم استدعاء التابع open() مرة ثانية.

الكود التالي هو من برنامج التشفير باستخدام شيفرة التحويل وهو يقوم بقراءة محتوى الملف الذي يكون اسمه مخزن في المتغير inputFilename

```
transpositionFileCipher.py
    # Read in the message from the input file
    fileObj = open(inputFilename)
    content = fileObj.read()
    fileObj.close()
```

الكتابة في الملفات:

لقد قمنا بقراءة الملف الأصلي، الآن سوف نقوم بكتابة النص المُشفر في ملف أخر وذلك من خلال استخدام الطريقة write()

أو يمكنك القيام بنفس العملية من خلال فتح الملف باستخدام نمط الكتابة 'w' بدلاً من نمط القراءة 'r' كما في المثال التالي:

```
>>> fo = open("c:\\spam.txt", 'w')
>>>
```

بالإضافة إلى نمطي القراءة والكتابة يوجد أيضاً نمط الإضافة "append" وهو مشابة لنمط الكتابة باستثناء أنه في نمط الإضافة فإن أي سلسلة نصية يتم كتابتها في النلف سوف تضاف إلى نهاية محتوى الحالي للملف

نمط الإضافة لا يقوم بالكتابة فوق المحتوى الحالي للملف.

من أجل فتح الملف في نمط الإضافة قم بكتابة 'a' كبرامتر ثاني للطريقة open()

بشكل مماثل لنمط القراءة عندما كنا نقوم بكتابة 'r' من أجل فتح الملف في نمط القراءة.

وفي حال عدم كتابة أي حرف كبرامتر ثاني في open() ستام فتح الملف بنمط القراءة بشكل افتراضي.

الطريقة writr() الخاصة ب file object:

يمكنك كتابة نص في داخل الملف من خلال استدعاء الطريقة write() method الخاصة ب file object

ولكن يجب أن تقوم بفتح الملف أولاً بنمط الكتابة وإلا سوف تحصل على رسالة الخطأ التالية

```
"io.UnsupportedOperation: not readable"
```

الطريقة ()write تأخذ برامتر واحد وهو السلسلة النصية التي تريد كتابتها داخل الملف

السطور من 43 إلى 45 هي لفتح الملف في نمط الكتابة والكتابة في الملف ومن ثم إغلاق هذا الملف.

```
transpositionFileCipher.py
42.     # Write out the translated message to the output file.
43.     outputFileObj = open(outputFilename, 'w')
44.     outputFileObj.write(translated)
45.     outputFileObj.close()
```

الآن وبعد أن تعرفنا على أساسيات القرءاة والكتابة في الملفات لنعد إلى الكود البرمجي لبرنامج تشفير الملفات بشيفرة التحويل.

كيف يعمل هذا البرنامج:

```
transpositionFileCipher.py
 1. # Transposition Cipher Encrypt/Decrypt File
 2. # http://inventwithpython.com/hacking (BSD Licensed)
 3.
 4. import time, os, sys, transpositionEncrypt, transpositionDecrypt
 5.
```

```
 6. def main():
 7.     inputFilename = 'frankenstein.txt'
 8.     # BE CAREFUL! If a file with the outputFilename name already exists,
 9.     # this program will overwrite that file.
10.     outputFilename = 'frankenstein.encrypted.txt'
11.     myKey = 10
12.     myMode = 'encrypt' # set to 'encrypt' or 'decrypt'
```

في السطر الرابع استخدمنا التعليمة import من أجل استيراد البرامج

transpositionEncrypt.py and *transpositionDecrypt.py*

ومن أجل استيراد الموديولات *time, os* and *sys* module

تعريف التابع ()main والذي سوف يقوم بتنفيذ كل العمليات في البرنامج عند استدعاءه.

المتغير inputFilename يحوي على سلسلة نصية وهي اسم الملف الذي سيتم القراءة النص منه ومن ثم يتم كتابة النص بعد عملية التشفير أو فك التشفير في الملف outputFilename

شيفرة التحويل تستخدم عدد صحيح وهي قيمة مفتاح التشفير ويتم تخزينها في المتغير myKey

إذا تم تخزين القيمة 'encrypt' في المتغير myMode فإن البرنامج سوف يقوم بتشفير محتوى الملف الموجود في inputFilename

وإذا تم تخزين القيمة 'decrypt' في المتغير myMode فإن البرنامج سوف يقوم بفك تشفير محتوى الملف الموجود في inputFilename

التابع ()os.path.exists:

قراءة الملفات هو أمر طبيعي وغير مؤذي ولكن يجب أن تكون حذرين عند الكتابة في الملفات.

إذا قمنا باستدعاء التابع ()open في نمط الكتابة وكان اسم الملف موجود مسبقاً فإن الملف الأول سيتم حذفه من أجل استخدام هذا الاسم للملف الثاني.

ويمكن أن يتم بالصدفة حذف ملف مهم إذا قمنا بتمرير اسمه إلى التابع ()open .

استخدام التابع ()os.path.exists يمكننا من فحص فيما إذا كان يوجد ملف بنفس الاسم.

التابع ()os.path.exists يقبل برامتر واحد وهو اسم الملف ويقوم بإعادة True إذا كان يوجد ملف بنفس الاسم و Fulse إذا لم يكن يوجد ملف بنفس الاسم.

التابع ()os.path.exists موجود في الموديول path module والتي هي موجودة في الموديول os module وعند استيراء الموديول os فإن الموديول path سيتم استيراده بشكل تلقائي أيضاً.

جرب المثال التالي في الشيل التفاعلية:

```
>>> import os
>>> os.path.exists('abcdf')
False
```

```
>>> os.path.exists('C:\\Windows\\System32\\calc.exe')
True
>>>
```

بالتأكيد سوف تحصل على هذه النتيجة في حال كنت تستخدم البايثون على نظام ويندوز لأن الملف calc.exe غير موجود في نظام لينكس.

```
transpositionFileCipher.py
    # If the input file does not exist, then the program terminates early.
    if not os.path.exists(inputFilename):
        print('The file %s does not exist. Quitting...' % (inputFilename))
        sys.exit()
```

قمنا باستخدام التابع ()os.path.exists من أجل فحص فيما إذا كان اسم الملف الموجود في المتغير inputFilename موجود مسبقاً وإلا سيتم عرض رسالة للمستخدم والخروج كمن البرنامج.

الطرق النصية ()startswith() and endswith:

```
transpositionFileCipher.py
    # If the output file already exists, give the user a chance to quit.
    if os.path.exists(outputFilename):
        print('This will overwrite the file %s. (C)ontinue or (Q)uit?' %
(outputFilename))
        response = input('> ')
        if not response.lower().startswith('c'):
            sys.exit()
```

إذا كان الملف الذي سوف يقوم البرنامج بالكتابة به موجود مسبقاً فسوف يطلب من المستخدم إدخال "C" في حال كان يريد المتابعة أو "Q" من أجل الخروج من البرنامج.

السلسلة النصية في المتغير response تقوم باستدعاء الطريقة ()lower والسلسلة النصية المُعادة من ()lower تقوم باستدعاء الطريقة النصية ()startswith

الطريقة النصية ()starswith ستعيد True إذا كانت السلسلة النصية التي يتم تمريرها لها موجودة في بداية السلسلة

جرب المثال التالي في الشيل التفاعلية:

```
>>> 'hello'.startswith('h')
True
>>> 'hello world'.startswith('h')
True
>>> 'hello'.startswith('H')
False
>>> spam = 'Jameel'
>>> spam.startswith('Jam')
True
```

في السطر 23 إذا لم يقم المستخدم بكتابة 'C' or 'countinue' , 'c' أو أي سلسلة نصية تبدأ بالحرف c سوف يتم استدعاء sys.exit() والخروج من البرنامج

المستخدم ليس مضطر إلى أن يكتب 'Q' من أجل الخروج من البرنامج لأن أي سلسلة نصية يقوم بكتابتها ولا تبدأ بالحرف 'C' سوف يتم استدعاء التابع sys.exit() من أجل الخروج من البرنامج.

الطريقة النصية endswith() تستخدم من أجل فحص فيما إذا كانت السلسلة النصية موجودة في نهاية سلسلة نصية أخرى

جرب المثال التالي في الشيل التفاعلية:

```
>>> 'Hello world'.endswith('world')
True
>>> 'Hello world'.endswith('Hello')
False
>>>
```

الطريقة النصية title():

بشكل مشابه للطرق النصية lower() and upper() والتي تعيد السلسلة النصية بحروف صغيرة أو كبيرة فإن الطريقة النصية title() تعيد السلسة النصية أول حرف كبير من كل كلمة وباقي الأحرف صغيرة.

جرب المثال التالي في الشيل التفاعلية:

```
>>> 'hello'.title()
'Hello'
>>> 'HELLO'.title()
'Hello'
>>> 'hELLO'.title()
'Hello'
>>> 'hello world HOW ARE YOU'.title()
'Hello World How Are You'
>>>
```

```
transpositionFileCipher.py
26.     # Read in the message from the input file
27.     fileObj = open(inputFilename)
28.     content = fileObj.read()
29.     fileObj.close()
30.
31.     print('%sing...' % (myMode.title()))
```

في السطور 27 إلى 29 يتم فتح الملف الذي يكون اسمه موجود في المتغير inputFilename وقراءة محتوى هذا الملف وتخزينه في المتغير conteny

في السطر 31 يتم عرض رسالة تخبر المستخدم أن عملية التشفير أو فك الشتفير سوف تبدأ ويجب أن يجوي المتغير myMode على السلسة النصية 'encrypt' or 'decrypt' والتي تقوم باستدعاء الطريقة النصية title() وسوف تظهر 'Encrypting...' or 'Decrypting...'

الوحدة time والتابع time.time():

كل أجهزة الحاسب تملك ساعة من أجل تتبع التوقيت والتأريخ.

برنامج البايثون يمكنه الوصول إلى هذه الساعة من خلال التابع time.time() (التابع time() موجود ضمن الموديول time)

التابع time.time() يعيد قيمة حقيقية float value لعدد الثواني من أول أيام الشهر الأول لعام 1970 هذه اللحظة تسمى عصر يونكس Unix Epoch

جرب المثال التالي:

```
>>> time.time()
1446316653.670954
>>> time.time()
1446316666.499506
>>>
```

القيمة الحقيقية التي يعيدها التابع time.time() غير مفيدة لتحديد قيمة الوقت الحالي ولكنها مفيدة من أجل المقارنة للفترة الزمنية بين استدعاء هذا التابع لأكثر من مرة

ويمكن الاستفادة من هذا التابع من أجل تحديد الفترة الزمنية التي استمر بها البرنامج بالعمل.

transpositionFileCipher.py

```
33.     # Measure how long the encryption/decryption takes.
34.     startTime = time.time()
35.     if myMode == 'encrypt':
36.         translated = transpositionEncrypt.encryptMessage(myKey, content)
37.     elif myMode == 'decrypt':
38.         translated = transpositionDecrypt.decryptMessage(myKey, content)
39.     totalTime = round(time.time() - startTime, 2)
40.     print('%sion time: %s seconds' % (myMode.title(), totalTime))
```

نريد معرفة الفترة الزمنية التي استمرت بها عملية التشفير أو فك التشفير.

في الأسطر 35 to 38 يتم استدعاء encryptMessage() or decryptMessage() وذلك بحسب القيمة المخزنة في المتغير myMode إن كانت 'encrypt' or 'decrypt'

قبل تنفيذ هذا الكود قمنا باستدعاء التابع time.time() وقمنا بتخزين قيمة الوقت الحالة في المتغير startTime

في السطر 39 وبعد استدعاء تابع التشفير أو فك التشفير نقوم باستدعاء التابع time.time() مرة ثانية وقمنا بطرح القيمة المحفوظة في المتغير startTime من القيمة المُعادة من استدعاء التابع للمرة الثانية و هذا سوف يعطينا عدد الثواني بين استدعائين لهذا التابع.

القيمة المعادة من حساب العبارة الجبرية time.time() – startTime يتم تمريرها إلى التابع round() والذي يقوم بتقريب القيمة الحقيقية إلى أقرب فاصلة عشرية و هذه القيمى يتم حفظها في المتغير totalTime

في السطر 40 يتم استدعاء التابع print() من أجل إظهار مدة الوقت التي استمرت به عملية التشفير أو فك التشفير.

```
                                                  transpositionFileCipher.py
42.     # Write out the translated message to the output file.
43.     outputFileObj = open(outputFilename, 'w')
44.     outputFileObj.write(translated)
45.     outputFileObj.close()
```

محتوى الملف الذي تم تشفيره أو فك تشفيره يتم حفظه في المتغير translated ولكن هذه القيمة يمكن أن تنسى عندما ينتهي البرنامج لذلك سوف نقوم بكتابة هذه السلسلة النصية في ملف وحفظه في القرص الصلب.

التعليمات في الأسطر 43 to 45 تقوم بفتح ملف جديد من (من خلال تمرير 'w' للتابع open())ومن ثم استدعاء الطريقة write()

```
                                                  transpositionFileCipher.py
47.     print('Done %sing %s (%s characters).' % (myMode, inputFilename,
len(content)))
48.     print('%sed file is %s.' % (myMode.title(), outputFilename))
```

```
49.
50.
51. # If transpositionCipherFile.py is run (instead of imported as a module)
52. # call the main() function.
53. if __name__ == '__main__':
54.     main()
```

بعد ذلك نقوم بطباعة بعض الرسائل للمستخدم تخبره أن العملية تمت وتخبره باسم الملف الذي تم الكتابة به.

السطر 48 هو آخر سطر في التابع main()

السطرين 53 and 54 يتم تنفيذهما بعد التعليمة def الموجودة في السطر 6 وسيتم استدعاء التابع main() إذا تم تشغيل هذا البرنامج ولم يتم استيراده من قبل برنامج آخر (في الفصل الثامن قمنا بشرح القيمة الخاصة __name__)

الخلاصة:

مبروك...

باستخدام هذا البرنامج والتوابع يمكنك تشفير الملفات النصية الموجودة في القرص الصلب في جهازك ومهما كان حجمها.

يوجد العديد من قيم المفاتيح المحتملة من أجل القيام بهجوم القوة الغاشمة brute-force على الرسالة المشفرة بشيفرة التحويل.

ولكن إذا قمنا بكتابة برنامج يتعرف على اللغة الانجليزية (سلسلة من الكلمات المفهومة) يمكن لجهاز الكمبيروتر أن يقوم بفحص خرج آلاف محاولات فك التشفير ليقرر ماهي قيمة المفتاح التي يتم بها فك التشفير بشكل صحيح ونحصل على رسالة مفهومة باللغة الانجليزية.



كتابة برنامج لإكتشاف اللغة الانجليزية

محتوى هذا الفصل:

- القواميس
- الطريقة ()split
- القيمة None
- الأخطاء "Divide by Zero"
- التوابع ()float(), int() and str
- الطريقة append الخاصة بالقوائم List
- البارامترات الأفتراضية
- حساب النسبة المئوية

الرسالة المشفرة باستخدام شيفرة التحويل يمكن أن يتم تشفيرها باستخدام الآلاف من مفاتيح التشفير.

جهاز الكمبيروتر يمكنه وبسهولة تطبيق هجوم القوة الغاشمة على هذه المفتايح ولكن يجب عليك بعدها أن تنظر إلى آلاف النصوص بعد عملية فك التشفير لتجد النص الصحيح.

هذه مشكلة كبيرة في هجموم القوة الغاشمة من أجل كسر شيفرة التحويل.

عندما يقوم الكمبيروتر بفك تشفير الرسالة باستخدام قيمة قيمة مفتاح خاطئة فالنتيجة ستكون هي نص صريح غير مفهوم.

نحتاج لبرنامج يجعل الكمبيروتر قادر على إدراك فيما إذا النص الصريح هو نص غير مفهوم او انه نص مفهوم باللغة الانجليزية.

وبهذه الطريقة إذا قام الكمبيوتر بفك تشفير الرسالة بالمفتاح الخطأ فهو سوف يعرف ان قيمة هذا المفتاح خاطئة وسوف ينتقل إلى قيمة مفتاح أخرى ويتابع عملية فك التشفير إلى أن يحصل على نص صريح ومفهوم باللغة الانجليزية عندها يتوقف ويقدم هذا المفتاح لمحلل الشيفرة.

وعندها لن يكون على محلل الشيفرة النظر إلى آلاف الرسائل بعد عملية فك التشفير.

كيف يمكن للكمبيروتر أن يفهم اللغة الانجليزية:

لا يمكنه.

على الأقل ليس بنفس الطريقة التي يستطيع الأنسان أن يدرك بها

الكمبيوتر لا يستطيع أن يفهم الرياضيات أو الشطرنج أو اللغات بل هو يقوم فقط بتنفيذ التعليمات.

ولكن هذه بالتعليمات يمكنها أن تحاكي السلوك المستخدم لحل مسائل الرياضيات أو لعب الشطرنج أو فهم اللغات.

ماهو التابع الذي نحتاج للقيام بهذه المهمة (لنسميه ()isEnhlish) سنقوم بتمرير قيمة نصية لهذا التابع وهو يعيد True إذا كانت هذه السلسلة هي نص باللغة الانجليزية ويعيد Fulse إذا كانت السلسلة النصية عبارة عن كلمات مكونة من أحرف عشوائية.

لنأخذ بعض الأمثلة على نصوص باللغة الانجليزية ونصوص لكلمات عشوائية غير مفهومة.

```
Robots are your friends. Except for RX-686. She will try to eat you.

ai-pey  e. xrx ne augur iirl6 Rtiyt fhubE6d hrSei t8..ow eo.telyoosEs  t
```

أول أمر يمكن أن نلاحظه هو أن نص اللغة الانجليزية مكون من الكلمات التي يمكن أن تكون موجودة في المعجم أما النص العشوائي لا يحوي على كلمات موجودة في المعجم

إذا قمنا بتقسيم النص إلى كلمات باستخدام الطريقة النصية split() عندها يمكننا البحث فيما إذا كانت هذه الكلمات موجودة في المعجم أو أنها كلمات عشوائية.

باستخدام الكود التالي:

```
>>> if word == 'aardvak' or word == 'abacus' or word == 'abandon' or .....
```

للقيام بهذه العملية بشكل اتوماتيكي سوف نستخدم ملف نصي يحوي على كلمات المعجم الانجليزية dictionary file

وسوف نقوم بكتابة تابع يقوم بفحص فيما إذا كانت هذه الكلمات موجودة في إحدى هذه الملفات.

dictionary file هو عبارة عن ملف نصي كبير يحوي على عدد كبير من الكلمات الانجليزية.

ليست كل الكلمات موجودة في "dictionary file" مثلاً يمكن أن يحوي النص على كلمات مثل "RX-686"

كما يمكن أن يحوي النص العشوائي على بعض الكلمات الانجليزية الصحيحة

التابع الذي سوف نقوم بكتابته لن يكون سهلاً، إذا كانت معظم الكلمات هي كلمات انجليزية صحيحة يمكننا أن القول أن هذا النص هو نص انجليزي صحيح و هذا يجعل احتمال أن يكون قد تم فك التشفير باستخدام المفتاح الصحيح هو احتمال كبير.

يمكنك تحميل هذا الملف (والذي يحوي على أكثر من 45,000 كلمة) من الرابط التالي:

http://invpy.com/dictionary.txt

التابع الذي سوف نقوم بكتابته isEnglish() سوف يقوم بتقسيم السلسلة النصية بعد عملية فك التشفير إلى كلمات ويقوم بفحص فيما إذا كانت هذه الكلمات هي كلمات باللغة الانجليزية أم أنها كلمات عشوائية.

إذا كانت النسبة الأكبر من هذه الكلمات هي كلمات من اللغة الانجليزية فهذا يعني أنه قد تم فك التشفير باستخدام مفتاح التشفير الصحيح.

:Detect English Module

البرنامج detectEnglish.py الذي سوف نقوم بكتابته في هذا الفصل لا يستطيع العمل بمفرده وسوف نقوم باستيرداه في برنامج التشفير لنتمكن من استدعاء التابع detectEnglish.isEnglish()

وهذا هو سبب عدم وجود التابع main() داخل هذا البرنامج.

الكود البرمجي لوحدة اكتشاف اللغة الانجليزية:

قم بفتح نافذة جديدة من محرر النصوص وقم بكتابة الكود التالي ثم قم بحفظه باسم detectEnglish.py ثم اضغط F5 من أجل تنفيذ البرنامج

Source code for detectEnglish.py

```python
# Detect English module
# http://inventwithpython.com/hacking (BSD Licensed)

# To use, type this code:
#   import detectEnglish
#   detectEnglish.isEnglish(someString) # returns True or False
# (There must be a "dictionary.txt" file in this directory with all English
# words in it, one word per line. You can download this from
# http://invpy.com/dictionary.txt)
UPPERLETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
LETTERS_AND_SPACE = UPPERLETTERS + UPPERLETTERS.lower() + ' \t\n'

def loadDictionary():
    dictionaryFile = open('dictionary.txt')
    englishWords = {}
    for word in dictionaryFile.read().split('\n'):
        englishWords[word] = None
    dictionaryFile.close()
    return englishWords

ENGLISH_WORDS = loadDictionary()


def getEnglishCount(message):
    message = message.upper()
    message = removeNonLetters(message)
    possibleWords = message.split()

    if possibleWords == []:
        return 0.0 # no words at all, so return 0.0

    matches = 0
    for word in possibleWords:
        if word in ENGLISH_WORDS:
            matches += 1
    return float(matches) / len(possibleWords)


def removeNonLetters(message):
    lettersOnly = []
    for symbol in message:
        if symbol in LETTERS_AND_SPACE:
            lettersOnly.append(symbol)
    return ''.join(lettersOnly)


def isEnglish(message, wordPercentage=20, letterPercentage=85):
    # By default, 20% of the words must exist in the dictionary file, and
    # 85% of all the characters in the message must be letters or spaces
    # (not punctuation or numbers).
    wordsMatch = getEnglishCount(message) * 100 >= wordPercentage
    numLetters = len(removeNonLetters(message))
    messageLettersPercentage = float(numLetters) / len(message) * 100
    lettersMatch = messageLettersPercentage >= letterPercentage
    return wordsMatch and lettersMatch
```

كيف يعمل هذا البرنامج:

detectEnglish.py

```
1. # Detect English module
2. # http://inventwithpython.com/hacking (BSD Licensed)
3.
4. # To use, type this code:
5. #   import detectEnglish
6. #   detectEnglish.isEnglish(someString) # returns True or False
7. # (There must be a "dictionary.txt" file in this directory with all English
8. # words in it, one word per line. You can download this from
9. # http://invpy.com/dictionary.txt)
```

التعليقات في بداية البرنامج تعطي إرشادات للمبرمج عن كيفية استخدام هذه الوحدة module

يجب أن تقوم بوضع الملف dictionary.txt في نفس المجلد الذي يحوي على detectEnglish.py وإلا لن يعمل هذا البرنامج

يمكنك تحميل هذا الملف من الرابط http://invpy.com/dictionary.txt

detectEnglish.py

```
10. UPPERLETTERS = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ'
11. LETTERS_AND_SPACE = UPPERLETTERS + UPPERLETTERS.lower() + ' \t\n'
```

في السطر 10 and 11 قمنا بإعداد بعض المتغيرات والتي تحوي على قيم ثابتة

المتغير UPPERLETTERS يحوي على الأحرف الكبيرة والمتغير LETTERS_AND_SPACE يحوي على الأحرف الكبيرة والأحرف الصغيرة بالإضافة إلى المسافة الفارغة والمسافة التي يخلقها الزر tab (\t) والانتقال لسطر جديد (\n).

detectEnglish.py

```
13. def loadDictionary():
14.     dictionaryFile = open('dictionary.txt')
```

يجب أن نقوم بفتح الملف dictionary وحفظ محتواه في متحول كسلسلة نصية ليتمكن البايثون من التعامل معه.

في البداية سنقوم باستدعاء open() من أجل فتح هذا الملف

قبل التعرف كود التابع ()loadDictionary سوف نتعرف على نوع البيانات dictionary data type

:Dictionary Data Type

هذا النوع من البيانات يحوي على قيم والتي يمكن أن تحوي على قيم متعددة أخرى، بشكل مشابه للقوائم lists

في نوع المتغير القائمة list كنا نستخدم قيمة عدد صحيح للفهرسة من أجل استدعاء القيم الموجودة في القائمة، مثل [42]spam

لكل مادة في المتغر من نوع القاموس dictionary يوجد مفتاح يستخدم لإستدعاء هذه المادة

(القيم المحفوظة في داخل القوائم والقواميس تسمى مواد items)

المفتاح يمكن أن يكون عدد صحيح أو فيمة نصية مثل spam['hello'] or spam[42]

القواميس تسمح لنا بتنظيم بيانات البرنامج بشكل أفضل من القوائم.

في القوائم نستخدم الأقواس المربعة [ ] أما في القواميس فنستخدم الأقواس الكبيرة { }

```
>>> emptyList = []
>>> empteDictionart = {}
```

المتغير من نوع قاموس تستخدم زوج من المفاتيح والتي يتم الفصل بينها باستخدام ' : ' ويتم الفصل بين المفاتيح المتعددة باستخدام ' , '

من أجل استدعاء قيمة من القاموس نستخدم الأقواس المربعة مع قيم المفاتيح المطلوبة، جرب المثال التالي في الشيل التفاعلية:

```
>>> spam = {'key1':'This is a value' , 'key2':42}
>>> spam['key1']
'This is a value'
>>> spam['key2']
42
>>>
```

جرب المثال التالي:

```
>>> spam = {'hello':42}
>>> eggs = spam
>>> eggs['hello'] = 99
>>> eggs
{'hello': 99}
>>> spam
{'hello': 99}
>>>
```

تعديل وإضافة مواد إلى القاموس:

يمكنك إضافة أو تعديل القيم في القاموس من خلال الفهرسة، جرب المثال التالي:

```
>>> spam = {42:'hello'}
>>> print(spam[42])
hello
>>> spam[42] = 'goodbye'
>>> print(spam[42])
goodbye
>>>
```

بما أن القوائم يمكن أن تحوي على قوائم أخرى في داخلها فإن القواميس أيضاً يمكن أن تحوي على قواميس أخرى في داخلها، جرب المثال التالي:

```
>>> spam = {42:'hello'}
>>> print(spam[42])
hello
>>> spam[42] = 'goodbye'
>>> print(spam[42])
goodbye
>>>
>>> foo = {'fizz': {'name': 'Ali', 'age': 144}, 'moo':['a', 'brown', 'cow']}
>>> foo['fizz']
{'name': 'Ali', 'age': 144}
>>> foo['fizz']['name']
'Ali'
>>> foo['moo']
['a', 'brown', 'cow']
>>> foo['moo'][1]
'brown'
>>>
```

استخدام التابع ()len مع القواميس:

التابع ()len يخبرنا بعدد المواد الموجودة في القائمة أو عدد الأحرف الموجودة في سلسلة نصية وهو أيضاً يخبرنا بعدد المواد الموجودة في القاموس

جرب المثال التالي:

```
>>> spam = {}
>>> len(spam)
0
>>> spam['name'] = 'Ali'
>>> spam['age'] = 30
>>> len(spam)
2
>>> .
```

استخدام المعامل in في القواميس:

المعامل in يستخدم لمعرفة وجود قيمة مفتاح معين في القاموس (قيمة المفتاح وليس قيمة معينة)، جرب المثال التالي:

```
>>> eggs = {'foo': 'milk', 'bar': 'bread'}
>>> 'foo' in eggs
True
>>> 'blah blah' in eggs
False
>>> 'milk' in eggs
False
>>> 'bar' in eggs
True
>>> 'bread' in eggs
False
>>> .
```

استخدام حلقة for مع القواميس:

يمكننا المرور على كل المفاتيح باستخدام حلقة for ، جرب المثال التالي:

```
>>> spam = {'name': 'Ali' , 'age': 40}
>>> for k in spam:
        print(k)
        print(spam[k])
        print('==========')


name
Ali
==========
age
40
==========
>>>
```

الفرق بين القواميس والقوائم:

القواميس تشبه القوائم في عدة أمور ولكن يوجد بعض الاختلافات وهي:

١- المواد في القواميس لا تحوي على أي ترتيب، لا يوجد المادة الأول والمادة الأخيرة كما في القوئم.
٢- القواميس لا يمكن جمعها باستخدام المعامل ' + ' إذا كنت تريد إضافة مواد جديدة إلى القاموس يجب عليك إدخال قيمة فهرسة لمفتاح جديد، مثلاً:
foo['a new key'] = 'a string'
٣- القوائم تتعامل فقط مع قيم الفهرسة التي تبدأ من الصفر إلى طول القائمة منقوص منه واحد أما القواميس فيمكنها التعامل مع أي قيمة لمفتاح الفهرسة
إذا كان لديك قاموس يخزن القيم في المتغير spam فيمكنك تخزين قيمة في spam[3] دون الحاجة لتخزين قيم في spam[1] or spam[2]

يجب أن تميز بين ملف القاموس الذي يحوي على كلمات المعجم والذي سوف نستخدمه في هذا البرنامج وبين المتغير من نوع قاموس في لغة البايثون.

إيجاد المواد في القواميس:

```
detectEnglish.py
15.     englishWords = {}
```

في التابع ()loadDictionary سوف نقوم بتخزين كل الكلمات الموجودة في الملف النصي "dictionary file" (كلمات المعجم) في متغير من نوع قاموس dictionary value (python data type)

للآسف نفس الاسم (قاموس) يجب أن تميز بينهما.

سوف نستخدم قاموس من أجل تخزين القيم النصية لكل كلمة في ملف القاموس dictionary file والسبب من ذلك هو أن المعامل ' in ' يعمل في القواميس بشكل أسرع من القوائم

تخيل أن لدينا القائمة والقاموس التاليين:

```
>>> listVal = ['spam', 'eggs', 'bacon']
>>> dictionaryVal = {'spam':0, 'eggs':0, 'bacon':0}
>>>
```

البايثون يستطيع تنفيذ التعليمة 'bacon' in dictionaryVal بشكل أسرع من التعليمة 'bacon' in listVal

وفي الوحدة detectEnglish module سيكون لدينا آلاف المواد وسوف يتم تنفيذ التعليمة word in ENGLISH_WORDS آلاف المرات عند استدعاء التابع isEnglish()

الطريقة split():

الطريقة النصية split() تقوم بإعادة قائمة لعدة سلاسل نصية

لمعرفة كيفية عمل هذه الطريقة جرب المثال التالي:

```
>>> 'I am a security engineer'.split()
['I', 'am', 'a', 'security', 'engineer']
>>>
```

كما يمكن استخدام هذه الطريقة بشكل مختلف كما في المثال التالي:

```
>>> 'helloxxxworldxxxhowxxxarexxxyou?'.split('xxx')
['hello', 'world', 'how', 'are', 'you?']
>>>
```

```
detectEnglish.py
16.     for word in dictionaryFile.read().split('\n'):
```

في السطر 16 استخدمنا حلقة for لتقوم بضبط قيمة المتغير word لكل قيمة في القائمة التي ستعيدها التعليمة *dictionaryFile.read().split('\n')* ، لنقم بتجزئة هذه العملية

dictionaryFile هو متغير يتم تخزينه في file object

استدعاء الطريقة dictionaryFile.read() سوف يقوم بقراءة كامل الملف ويعيده على شكل سلسلة نصية طويلة جداً.

سوف نقوم باستدعاء الطريقة split() من أجل الفصل بين هذه الكلمات.

وبالتالي فإن التعليمة dictionaryFile.read().split('\n') سوف تقوم بإعادة قائمة تحوي على الكلمات الموجودة في هذا الملف.

القيمة None:

هي عبارة عن قيمة خاصة يمكن اسنادها لمتغير معين

وهي القيمة الوحيدة لنوع البيانات None (بشكل مشابه لنوع البيانات المنطقي والذي يملك قيمتين: الصفر والواحد) نوع البيانات None يملك قيمة وحيدة وهي None

من المفيد استخدام هذه القيمة عندما تحتاج لاستخدام قيمة غير موجودة والقيمة None تكتب بدون إشارات تنصيص والحرف الأول منها يجب أن يكون حرف كبير.

مثلاً: إذا كان لديك متغير له الاسم quizAnswer والذي سوف يحوي على إجابة المستخدم True or False

يمكنك ضبط القيمة None للمتغير quizeAnswer في حال لم يقم المستخدم بالإجابة على هذا السؤال.

استدعاء التوابع التي لن تقوم بإعادة أي شئ (يتنهي التابع بدون استدعاء التعليمة return) يمكن أن تسند لها القيمة None

```
17.         englishWords[word] = None
```

في برنامجنا سوف نستخدم متغير من نوع قاموس dictionary باسم englishWords وسوف نستخدم المعامل in من أجل إيجاد المفاتيح في داخله.

لن نهتم للقيمة التي سوف تخزن في كل مفتاح، فقط سوف نستخدم القيمة None.

الحلقة for التي تبدأ في السطر 16 سوف تمر على كل كلمة في الملف النصي الذي يجوي على كلمات المقاموس والسطر 17 سوف نستخدم word كمفتاح في englishWords مع القيمة None التي ستكون مخزنة في كل مفتاح.

```
detectEnglish.py
    dictionaryFile.close()
    return englishWords
```

وبعد أن تنتهي الحلقة for فإن المتغير من نوع قاموس englishWords سوف يحوي على آلاف المفاتيح (الكلمات) وعندها سوف نقوم بإغلاق الملف النصي.

```
detectEnglish.py
21. ENGLISH_WORDS = loadDictionary()
```

في السطر 21 نستدعي التابع ()loadDictionary ونقوم بتخزين القيم التي يعيدها في المتغير ENGLISH_WORDS

```
detectEnglish.py
24. def getEnglishCount(message):
25.     message = message.upper()
26.     message = removeNonLetters(message)
27.     possibleWords = message.split()
```

التابع ()getEnglishCount سوف يأخذ سلسلة نصية واحدة ويعيد قيمة عدد حقيقي للدلالة على عدد الكلمات التي تعتبر كلمات باللغة الانجليزية.

القيمة 0.0 تعني أنه لا يوجد أي كلمة انجليزية في المتغير message

أما القيمة 1.0 تعني أنه كل الكلمات في المتغير message هي كلمات انجليزية

التابع ()getEnglisgCount سوف يعيد قيمة بين 1.0 – 0.0 والتابع ()isEnglish سوف يستخدم هذه القيمة لتحديد فيما إذا كان النص بعد عملية فك التشفير هو نص باللغة الانجليزية أو لا.

في البداية يجب أن نقوم بخلق قائمة تحوي على الكلمات الموجودة في السلسلة النصية المخزنة في المتغير message .

في السطر 25 سوف نقوم بتحويل الكلمات إلى أحرف كبيرة.

في السطر 26 سوف نحذف القيم الغير حرفية من السلسلة النصية مثل الأرقام وعلامات الترقيم وذلك من خلال استدعاء التابع ()removeNonLetters (سوف نشرح طريقة عمل هذا التابع لاحقاً)، وأخيراً سوف نستدعي الطريقة ()split في السطر 27 من أجل القيام بتقسيم السلسلة النصية إلى كلمات وحفظها في المتغير possibleWords.

مثلاً إذاقمنا بتمرير السلسة النصية 'Hello there. How are you?' إلى التابع getEnglishCount() فسوف يعيد القيم التالية ويخزنها في المتغير possibleWords

['HELLO' , 'THERE' , 'HOW' , 'ARE' , 'YOU']

```
detectEnglish.py
    if possibleWords == []:
        return 0.0 # no words at all, so return 0.0
```

إذا كان المتغير message يحوي على أرقام مثل '12345' فسوف يقوم التابع removeNonLetters() بحذفها ويعيد سلسلة نصية فارغة وعندها سوف يتم استدعاء الطريقة split() على سلسلة نصية فارغة والتي سوف تعيد قائمة فارغة وسوف يتم إعادة القيمة 0.0

```
detectEnglish.py
32.     matches = 0
33.     for word in possibleWords:
34.         if word in ENGLISH_WORDS:
35.             matches += 1
```

القيمة الحقيقية التي يعيدها التابع getEnglishCount() ستكون بين القيمتين 0.0 – 1.0

في البداية سوف نقوم بتعين القيمة 0 للمتغير matches ثم نستخدم حلقة for في السطر 33 من أجل أن تمر على كل كلمة موجودة في المتغير possibleWords لترى فيما إذا كانت هذه الكلمة موجودة في القاموس ENGLISH_WORDS

إذا كانت موجودة سوف يتم زيادة قيمة المتغير matches بمقدار واحد.

عندما تنتهي حلقة for من عملها سوف يتم تخزين عدد الكلمات الانجليزية في المتغير matches

البرنامج يعمل بالطريقة التالية: إذا كانت الكلمة موجودة في الملف النصي dictionary.txt فهي كلمة انجليزية وإذا لم تكن موجودة فهي كلمة عشوائية.

خطأ القسمة على صفر:

```
detectEnglish.py
36.     return float(matches) / len(possibleWords)
```

إعادة قيمة حقيقية بين 1.0 – 0.0 تتم من خلال قسمة عدد الكلمات المكتشفة على أنها كلمات انجليزية على عدد الكلمات الكلية.

هذه العملية تتم باستخدام المعامل " / " ، يجب أن تنتبه لكي لا نقوم بعملية القسمة على العدد صفر.

أبي رحمه الله قد علمني العبارة التالية: " القسمة على صفر هي كفر"

البايثون سوف يعيد رسالة خطأ عند محاولة القسمة على صفر، جرب المثال التالي:

```
>>> 42 / 0
Traceback (most recent call last):
  File "<pyshell#0>", line 1, in <module>
    42 / 0
ZeroDivisionError: division by zero
>>>
```

الحالة الوحيدة التي يمكن أن تسبب هذه النتيجة هي عندما تكون نتيجة len(possibleWords) هي صفر

في كود البرنامج وفي السطر 29 قمنا بفحص القائمة possibleWords إذا كانت قائمة فارغة فسوف يتم تنفيذ التعليمة الموجودة في السطر 30 ولن نحصل على خطأ القسمة على صفر في السطر 36

التوابع float() , int() and str():

```
detectEnglish.py
36.     return float(matches) / len(possibleWords)
```

القيمة المخزنة في المتغير matches هي قيمة عدد صحيح integer

قمنا بتمرير هذه القيمة إلى التابع float() والذي يعيد قيمة عدد حقيقي، جرب المثال التالي:

```
>>>
>>> float(42)
42.0
>>> int(42.0)
42
>>> int("42")
42
>>> str(42)
'42'
>>> str(42.7)
'42.7'
>>>
```

التوابع float(), int(), and str() مفيدة في تحويل القيم بين أنواع المتغيرات

```
detectEnglish.py
39. def removeNonLetters(message):
40.     lettersOnly = []
41.     for symbol in message:
```

في التابع getEnglishCount() قمنا باستدعاء التابع removeNonLetters() من أجل إعادة السلسلة النصية بدون أرقام أو علامات ترقيم.

الكود في التابع removeNonLetters() يبدأ بقائمة فارغة ثم يتم المرور على الأحرف الموجودة في المتغير message، إذا كانت الأحرف موجودة في السلسلة LETTERS_AND_SPACE فسوف يتم إضافتها إلى نهاية القائمة وإذا كانت المحارف هي أرقام أو علامات ترقيم فلن تكون موجودة في السلسلة LETTERS_AND_SPACE ولن يتم إضافتها إلى القائمة.

```
detectEnglish.py
42.         if symbol in LETTERS_AND_SPACE:
43.             lettersOnly.append(symbol)
```

في السطر 42 قمنا بفحص فيما إذا كان كل محرف موجود في السلسلة LETTERS_AND_SPACE وإذا كان موجود سيتم إضافته إلى نهاية القائمة lettersOnly باستخدام الطريقة append()

```
detectEnglish.py
44.     return ''.join(lettersOnly)
```

بعد أن تنتهي حلقة for الموجودة في السطر 41 من عملها سوف تحوي القائمة lettersOnly على الأحرف والفراغات فقط ومن أجل سلسلة نصية واحدة من هذه القائمة سوف نستدعي

الطريقة النصية ()join على سلسلة نصية فارغة والتي سوف تقوم بجمع كل السلاسل النصية الموجودة في القائمة lettersOnly مع بعضها.

```
detectEnglish.py
47. def isEnglish(message, wordPercentage=20, letterPercentage=85):
48.     # By default, 20% of the words must exist in the dictionary file, and
49.     # 85% of all the characters in the message must be letters or spaces
50.     # (not punctuation or numbers).
```

التابع ()isEnglish يقبل سلسلة نصية كدخل له ويعيد قيمة منطقية True or Fulse من أجل الدلالة إذا كانت هذه السلسة النصية هي نص باللغة الانجليزية أم أنها كلمات عشوائية.

في السطر 47 قمنا بتعريف التابع باستخدام ثلاث برامترات.

البرامترات wordPercentage and letterPercentage تحوي على قيم إفتراضية وإذا لم يقم التابع بتمرير قيم أخرى فسوف يتم استخدام هذه القيم بشكل افتراضي.

إذا تم إستدعاء التابع ()isEnglish مع برامتر واحد وهو السلسة النصية فعندها ستكون قيم البارمترات الأخرى هي القيمة الافتراضية

wordPercentage = 20

letterPercentage = 85

الجدول التالي يظهر بعد طرق استدعاء هذا التابع باستخدام قيم مختلفة

Function Call	Equivalent To
isEnglish('Hello')	isEnglish('Hello', 20, 85)
isEnglish('Hello', 50)	isEnglish('Hello', 50, 85)
isEnglish('Hello', 50, 60)	isEnglish('Hello', 50, 60)
isEnglish('Hello', letterPercentage=60)	isEnglish('Hello', 20, 60)

عندما يتم استدعاء التابع ()isEnglish بدون البرامترين الثاني والثالث فإن التابع سوف يطلب أن يكون 20% من الكلمات الموجودة في المتغير message هي كلمات انجليزية (موجودة في الملف النصي الي يحوي على كلمات القاموس) و 85% من المحارف في المتغير message هي أحرف.

حساب النسبة المئوية:

النسبة المئوية هي رقم بين 100- 0

السلسلة النصية التالية: "Hello cat Moose fsdkl ewpin" تحوي على خمس كلمات ولكن ثلاثة فقط من هذه الكلمات هي كلمات باللغة الانجليزية.

من أجل حساب النسبة المئوية لعدد الكلمات الأنجليزية نقم بقسمة عدد الكلمات الانجليزية على عدد الكلمات الكلية ومن ثم نضرب النتيجة بالعدد 100

$$3 / 5 * 100 = 6$$

النسبة المئوية هي عدد بين 0% (لا يوجد أي كلمة انجليزية) و 100% (كل الكلمات هي كلمات انجليزية)

التابع isEnglish() سوف يحدد النص إذا كان نص باللغة الانجليزية أو لا إذا كان يحوي على الأقل 20% من الكلمات الانجليزية الموجودة في ملف القاموس و 85% من المحارف هي سلاسل نصية أو أحرف أو فراغات.

```
detectEnglish.py
51.     wordsMatch = getEnglishCount(message) * 100 >= wordPercentage
```

في السطر 51 يتم حساب النسبة المئوية للكلمات الانجلزية في الرسالة من خلال تمرير المتغير message إلى التابع getEnglishCount() والذي يقوم بإعادة قيمة بين 0.0 and 1.0 ومن ثم يتم حساب النسبة المئوية بالإعتماد على هذه القيمة وذلك بالقيام بعملية الضرب بالعدد 100

إذا كان العدد الناتج أكبر أو يساوي قيمة البرامتير wordPercentage فسوف يتحقق الشرط ويتم حفظ القيمة True في المتغير wordsMatch وفي حال عدم تحقق الشرط سوف يتم حفظ القيمة False في المتغير wordsMatch

```
detectEnglish.py
    numLetters = len(removeNonLetters(message))
    messageLettersPercentage = float(numLetters) / len(message) * 100
    lettersMatch = messageLettersPercentage >= letterPercentage
```

في الأسطر 52 to 54 يتم حساب المسبة المئوية للأحرف في الرسالة وذلك من أجل تحديد النسبة المئوية لكل من الأحرف والفراغات.

ويتم ذلك من خلال تقسيم عدد الأحرف على عدد المحارف الكلية (أقصد بكلمة محارف الأحرف والفراغات وعلامات الترقيم).

في السطر 52 قمنا باستدعاء التابع removeNonLetters(message) والذي سوف يعيد سلسلة نصية تحوي على الأحرف وعلامات الترقيم وبدون فراغات.

في السطر 53 يتم حساب النسبة المئوية من خلال تحويل قيمة numLetters إلى قيمة عدد حقيقي ومن ثم القسمة على عدد المحارف الكلية في الرسالة وضرب الناتج بالقيمة 100

في السطر 54 يتم فحص النسبة المئوية إذا كانت أكبر أو مساوية لقيمة letterPercentage وتخزين النتيجة المنطقية True or False في المتغير lettersMatch

detectEnglish.py

```
55.     return wordsMatch and lettersMatch
```

التابع isEnglish() يعيد True فقط إذا كان كل من المتغيران wordsMatch and lettersMatch يحويان القيمة True لذلك قمنا بتمرير هذين المتغيرن ضمن عبارة منطقية لتنفيذ العملية المنطقية and

إذا كانت قيمة كل من المتغيران wordMatch and letterMatch هي True فإن التابع isEnglish() سوف يعيد القيمة True وهذا يعني أن الرسالة هي عبارة عن نص باللغة الانجليزية.